An Analytical Framework for Evaluation of Reliability and Security in Advanced Network Systems

Von der Fakultät für Elektrotechnik, Informationstechnik, Physik
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades einer Doktorin

der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von Anna Engelmann

aus Rubzowsk, Russische Föderation

eingereicht am 05.03.2020

mündliche Prüfung am 29.06.2020

1. Referentin: Prof. Admela Jukan, TU Braunschweig, Germany
2. Referentin: Prof. Muriel Médard, Massachusetts Institute of Technology, USA
3. Referent: Prof. Suresh Subramaniam, George Washington University, USA
4. Referentin: Prof. Regine Mallwitz, TU Braunschweig, Germany

Druckjahr: 2021

Dissertation an der Technischen Universität Braunschweig,
Fakultät für Elektrotechnik, Informationstechnik, Physik

Abstract

Today, anonymous networks such as The Onion Routing (Tor) have been designed to ensure anonymity, privacy and censorship prevention, which have become major concerns in modern society. Although the Tor network provides layered encryption and traffic tunneling against eavesdropping attacks, the jamming attacks and their impact on the network and network services can not be efficiently handled today. Moreover, to defy modern censorship, it is not enough just to use the Tor network to hide the client's identity and the message content as the censorship has become a type of jamming attack, which prevents users from connecting to the censored network nodes by blocking or jamming (Tor) traffic. In network security, the main tools to protect privacy and anonymity as well as integrity and service reliability against eavesdropping and jamming, respectively, are diversity, randomness, coding or encryption and over-provisioning, all less exploit in traditional networks.

This thesis provides radical new network concepts to address the needs of traditional networks for privacy, anonymity, integrity, and reliability; and designs *advanced network systems* based on parallel transmission, random routing, erasure coding and redundant configurations as tools to offer diversity, randomness, coding and over-provisioning. Since the network systems designed in this thesis can not be evaluated with existing analytical models due to their rather complex configurations, the main focus of this work is a development of novel analytical approaches for evaluation of network performance, reliability and security of these systems and to show their practicality. The provided analysis is based on combinatorics, probability and information theory. In contrast to current reliability models, the analysis in this thesis takes into account the sharing of network components, heterogeneity of software and hardware, and interdependence between failed components. The significant property of the new security analysis proposed is the ability to assess the level of privacy, anonymity, integrity and censorship success when multiple jamming and eavesdropping adversaries reside in the network.

Kurzfassung

Derzeit werden anonyme Internet Kommunikationssysteme, wie The Onion Routing (Tor), verwendet, um die Anonymität, die Privatsphäre und die Zensurfreiheit der Internetnutzer zu schützen. Obwohl das Tor-Netzwerk eine Mehrfachverschlüsselung und ein IP-Tunneling des Datenverkehrs gegen Lauschangriffe (Eavesdropping) bietet, kann ein beabsichtigtes Stören (Jamming) der Übertragung und den daraus resultierenden Auswirkungen auf die Netzwerkfunktionen derzeit nicht effektiv abgewehrt werden. Auch das moderne Zensurverfahren im Internet stellt eine Art des Jammings dar. Dabei werden Internetnutzer daran gehindert eine Verbindung zu bestimmten Webseiten und dem Tor Netzwerk herzustellen oder der Tor-Traffic wird durch zusätzliche Störungen stark beeinträchtigt und vom Empfänger verworfen. Deswegen kann das Tor Netzwerk zwar die Identität der Tor-Nutzer und die Inhalte ihrer Nachrichten geheim halten, die Internetzensur kann dadurch nicht verhindert werden. Um die Netzwerksicherheit und insbesondere Anonymität, Privatsphäre und Integrität zusammen mit der Verfügbar.- und Zuverlässigkeit von Netzwerkservices zu gewährleisten, sind Diversität, Zufallsprinzip, Codierung (auch Verschlüsselung) und eine Überversorgung, die in den konventionellen Netzwerksystemen eher sparsam angewendet werden, die wichtigsten Mittel gegen Security-Angriffe.

Diese Arbeit befasst sich mit grundlegend neuen Konzepten für Kommunikationsnetze, die einen Schutz der Anonymität und der Privatsphäre im Internet bei gleichzeitiger Sicherstellung von Integrität, Verfügbarkeit und Zuverlässigkeit ermöglichen. Die dabei verwendeten Konzepte sind die parallele Datenübertragung, das Random Routing, das Erasure Coding und redundante Systemkonfigurationen. Damit sollen Diversität, Zufallsprinzip, Codierung und eine Überversorgung gewährleistet werden. Da die entwickelten Übertragungssysteme komplexe Strukturen und Konfigurationen aufweisen, können existierende analytische Modelle nicht für eine fundierte Bewertung angewendet werden. Daher ist der Schwerpunkt dieser Arbeit neue analytische Verfahren für eine Bewertung von unter-

schiedlichen Netzwerkleistungsparametern, Zuverlässigkeit und Security zu entwickeln und die Praxistauglichkeit der in der Arbeit aufgeführten neuen Übertragungskonzepte zu beurteilen. Im Gegensatz zu existierenden Zuverlässigkeitsmodellen berücksichtigt der analytische Ansatz dieser Arbeit die Vielfalt von beteiligten Netzwerkkomponenten, deren komplexe Zusammenhänge und Abhängigkeiten im Fall eines Ausfalls. Eine wichtige Eigenschaft der entwickelten Security-Analysis ist die Fähigkeit unterschiedliche Security-Merkmale, z.B. die Wahrscheinlichkeit für eine erfolgreiche Internetzensur oder Deanonymisierung, bei gleichzeitiger Berücksichtigung von mehreren unterschiedlichen Sicherheitsangriffen zu bewerten.

Acknowledgments

This thesis would not have been possible without the help and support of many wonderful people. First, I would like to thank my PhD advisor Prof. Admela Jukan for her guidance and support throughout my doctorate study. Her visionary thinking, ability to view engineering problems from a practical standpoint, persistence and hard work have continually inspired me. She gave me the exceptional opportunity, freedom and resources to develop as a researcher and lead independent research activities within DFG and European research projects, for which I am truly appreciative. I would also like to thank the members of my thesis committee, Prof. Muriel Médard and Prof. Suresh Subramaniam for their valuable time to evaluate my work. Special thanks go to Prof. Muriel Médard for the research collaboration to investigate the potential of RLNC in Ethernet-over-Optical networks. I am also very grateful to Prof. Regine Mallwitz who generously accepted to chair the examination committee.

I was extremely fortunate to work with some magnificent people. My special thanks go to Dr. Wolfgang Bziuk. Thank you for sharing your time, your experiences, and your incredible expertise with me. I also would like to thank Prof. Xiaomin Chan for the mentorship at the beginning of this PhD adventure and for her friendship. I am very grateful to Dr. Sandeep Kumar Singh, Mounir Bensalem, Dr. Marcel Caria, Dr. Mohit Chamania, Jasenka Dizdarevic, Francisco Carpio, Prof. André Costa Drummond, Cao Vien Phung, Zied Ennaceur, Marek Drogon, Randa Zarrouk and Dr. Ayoub Mars who have been such wonderful colleagues and great friends. It has been a pleasure working with and learning from all of you. I am so grateful to Prof. Wael Adi and Dr. Saleh Mulhem for many constructive discussions on security topics. I would also like to thank the partners of the METERACOM and the SENDATE projects who have been excellent collaborators. Special thanks go to Ms. Christiane Geller and Mr. Marc Schmidtchen for all administrative support.

None of this would have been possible without the support of my family. Words will always be inadequate to thank my husband, André Uibel, and our wonderful son, Anton, for being a constant source of love and encouragement. I am also very grateful to much more people who were not mentioned here, thank you so much.

List of Figures

List of Tables

Contents

1

Introduction

Today, network security is defined through confidentiality, integrity, authentication, anonymity, and availability. The authentication process can be preserved and compromised at communication end-points only. In contrast, confidentiality, integrity, anonymity, and availability can be compromised on any link and in any node in the network and, thus, need particular attention. Confidentiality also referred to as communication privacy, and anonymity can be threatened by an eavesdropping attack, while data encryption can not prevent any adversary to track interactions and interaction patterns or perform traffic analysis. The data integrity can be interpreted as the ability of a communication system to send source data toward the destination, which should be able to recover this data despite any transmission impairment including jamming attack. The availability and reliability are often interchangeable and can be defined as a probability that a system can offer and complete the processing of the intended service. The service reliability can be affected by operation errors of the system due to non-malicious failures or jamming [12, 13, 14].

The main tools to improve the reliability and security in network systems in the presence of jamming or eavesdropping attacks are randomness, diversity, coding, and over-provisioning. The randomness is a tool to increase the level of diffusion and confusion to hamper the

prediction of system behavior. Diversity ensures that an attacker can only compromise some parts of the system or confidential data. The data coding and encryption protect or hide secret data content, respectively. The over-provisioning in the form of redundancy ensures integrity and reliability when some parts of data or system are lost or have failed, respectively. Thus, this thesis designs *advanced network systems* that inherently utilize parallel transmission, random routing, erasure coding, coding redundancy and network redundancy to provide diversity, randomness, coding, and over-provisioning; and investigates their reliability and security.

The parallel transmission is introduced in High-speed Ethernet standard IEEE802.3. Hence, the 40/100/400Gb/s Ethernet traffic can be packetized and distributed over multiple parallel lanes (e.g., 10 lanes x 10Gb/s, for 100Gb/s), referred to as Multi-Lane Distribution (MLD) [2]. Each lane can be mapped onto parallel output interfaces and optical channels, i.e., optical paths, for parallel transmission in Optical Transport Networks (OTN). Parallel transmission in optical networks in the form of multi-path routing is a well-studied subject, known, however, for a strict constraint on optimality [15, 16]. The optical paths need to be selected to minimize the inter-lane skew, i.e., the time difference between receiving the data on the first and the last lane, which limits its practical application. On IP Layer, Equal-Cost Multi-Path (ECMP) is used as the actual routing protocol in Data Center Networks (DCNs). ECMP does not differentiate between short and long flows, which compete for the same link, resulting in either congested or underutilized links. Thus, it has been proposed to break the long flows into parallel subflows and uniformly distribute them over the network, i.e., parallel transmission, to improve the network performance [17, 18, 19].

Random routing of parallel paths may not only increase security but also eliminate the need for complex paths optimization of multipath routing, required to minimize the inter-lane skew. Any skew

value requires the so-called *de-skewing* via buffering at the receiver. However, the High-speed Ethernet standard IEEE802.3 defines the maximum skew of 180 ns, which may be difficult to meet in random settings, resulting in an overload of deskew buffer and data being dropped at the receiver [2]. Thus, a comprehensive analysis of skew and the required deskew buffer is crucial to validate the practicality of random routing for advanced network systems.

An erasure coding is suitable as an adaptive data coding in addition to standard techniques such as encryption and Forward Error Correction (FEC) [20, 21, 22, 23]. In contrast to these standard solutions for confidentiality and integrity, the erasure coding can provide flexibility in combination with intelligent algorithms and adapt coding parameters to the changing state of the environment caused by system failures or security attacks. With coding redundancy added into data, their protection against losses is implemented by erasure correction preserving data integrity in the case of transmission impairments or jamming attacks. Here, each erroneous part of data can be replaced by coding redundancy eliminating the need for retransmission of lost bits. An erasure code can be easily applied to the parallelized traffic to reach a reliable transmission. For instance, the coded data from multiple paths, i.e., coding redundancy, can be sent on a few additional parallel paths, i.e., path redundancy, to perform erasure correction at the receiver [24]. In this case, integrity correlates with the reliability of data transmission, as both corrupted data and path failures can be recovered. Additionally, randomly chosen coding coefficients can be utilized to increase confidentiality, whereby any eavesdropper has to access coded data over all parallel paths and guess coding coefficients to reveal the data content.

Similar to coding and path redundancy for data integrity and transmission reliability, the over-provisioning with redundant system components, known as backup protection, can improve the service reliability in the case of any failure of this type of components

caused by a misconfiguration or jamming attacks. In IP networks, backup protection is a standard solution to protect services composed of different network functions such as Network Address Translation (NAT), firewall, and others. Today, Network Function Virtualization (NFV) decouples traditional network functions from the physical hardware. Hence, the network functions present software instances so-called Virtual Network Functions (VNFs) running on virtual machines (VM) hosted by physical devices. Thus, end-to-end service in NFV is a chain of VNFs referred to as Service Function Chain (SFC). SFC is affected by processing vulnerability and failures caused by hardware or software as a result of malfunction or security attacks, e.g., jamming [25]. However, the failure of any VNF disrupts the entire SFC. To increase SFC reliability, the overprovisioning in form of VNF backup protection utilizes additional VNF replicas. To be effectively used, the backup protection needs fast failure detection and rapid reaction to detected failures. However, the prompt replacement of failed VNFs trough backup VNFs is a challenging task, which requires fast VNF migration and traffic redirection to the backup VNFs [26]. As any VNF failure can result in traffic loss, the reliability of data transmission correlates with the reliability of SFC. Certainly, the application of a parallel transmission and erasure coding to provide the path and coding redundancy may have the potential to reduce the overhead of VNF migration and traffic redirection when coding redundancy is sent to backup VNFs over redundant paths. However, designing reliable SFC is not without issues as assessing SFC reliability and dimensioning of backup resources as a function of the VNF placement strategy is a complex problem. That is due to a need for consideration of co-location and sharing of hardware components, heterogeneity of software and hardware, and complex interdependency between them [27]. Thus, SFC reliability has not been addressed yet analytically observing all required parameters and is one of the focal points of this thesis.

Along with service reliability, anonymity, privacy, and censorship have become major concerns in modern society. The anonymity and privacy can be compromised by eavesdropping, while censorship can be implemented as jamming. Therefore, anonymous networks such as The Onion Routing (Tor) have been developed to provide practical mechanisms such as layered encryption and traffic tunneling ensuring user anonymity, privacy, and mitigating censorship [28, 29, 30, 31]. In general, Tor is designed to prevent censorship by hiding users' identity and message content. However, today's censorship prevents users from connecting to Tor by blocking the known public Onion Routers (OR) or by jamming Tor traffic [32]. As a censoring entity also collects and blocks the secret ORs, i.e., bridges, Tor does not protect against censoring entities themselves, which not only jams and blocks Tor traffic but also degrade the Tor performance. In absence of censorship, the application of multipath routing improves the Tor performance and randomize the tunnel distribution, whereby the bandwidth from different Tor circuits can be effectively aggregated [33, 34, 35, 36]. Censorship by jamming or circuit blocking, on the other hand, remains an issue in real networks.

This thesis designs and analyzes *advanced network systems*, which can exploit traffic and paths parallelism, random routing, erasure coding, and diverse redundancy. A novel generalized analysis based on combinatorics enables the evaluation of some essential performance parameters such as skew, deskew buffer, coding overhead, etc., as well as resulting reliability and security of designed advanced network systems. The random routing strategies are investigated on two forwarding methods: random circuit switching and random packet switching, whereby the performance of data transmissions over the optimized and randomly selected parallel paths with and without coding is analyzed and compared.

To increase SFC reliability and alleviate the challenges associated with complex VNF failure detection, time-consuming VNF migra-

tion, and traffic redirection in the case of backup VNF protection, a new VNF protection based on *systematic erasure coding* is investigated. Here, both failure protection methods (with coding and backup) are compared concerning SFC reliability and overall overhead. The novel generalized reliability analysis based on combinatorics and probability theory enables evaluation of the SFC reliability as a function of VNF placement strategies in generic inter- and intra- Data Center Networks (DCNs). The notable feature of the new reliability analysis is the ability to take into account failures of DC, racks, servers, VMs and path segments and, besides, sharing, heterogeneity, and interdependencies of system components.

For security analysis, in contrast to other studies, this thesis investigates different types of eavesdropping and jamming adversaries, that can access network links and nodes simultaneously and in different combinations, and compromise anonymity, privacy, integrity, and reliability of communication. A new combinatorial approach to security analysis allows evaluating the security level of advanced network systems with different configurations while taking into account diverse attacks and a different number of adversaries. When applying erasure coding, its effectiveness to balance the demands of data integrity, reliability and privacy is analyzed in the case of simultaneous jamming and eavesdropping. The probability of censorship success is assessed by observing a Tor network extended to parallel transmission, random routing, coding, and diverse redundancy.

Finally, the feasibility of anonymous optical networks to overcome the drawbacks of the traditional Tor is investigated. To enable all-optical layered encryption at a line rate, a novel all-optical stream cipher is designed and evaluated against practicability. We show that optical cryptosystems require new techniques for security assessment and validation. Thus, this thesis creates new security benchmarks based on information theory, which allow us engineering anonymous optical networks with a particular security level.

1.1 Thesis Contributions

This thesis contributes to several design, modeling and analysis aspects of advanced network systems. The main contribution of this thesis is a novel design technique and engineering of fundamentally new network systems with intrinsic randomness, diversity, data coding, and over-provisioning as tools to address the needs of modern networks for privacy, anonymity, integrity, and reliability. As a combination of randomness, diversity, coding, and over-provisioning lead to less investigated complex network configurations, the major part of this work provides novel analytical models and approaches for evaluation and benchmarking of reliability and security in that network systems. Here, reliable data transmission concerning data integrity and a reliable network function chaining is considered as a service, which can be disturbed due to system faults or security attacks. Hence, to assess service reliability, a novel reliability analysis based on combinatorics and probability theory is introduced and verified. In addition, a new security evaluation approach is presented, whereby, in contrast to previous studies, different types of attacks, their combinations and different amount of adversaries can be taken into account and analyzed. The information-theoretical security evaluation and benchmarking are proposed and enable the engineering of an anonymous optical network as a special case of a reliable and secure network implementation to provide randomness, diversity, data coding, and over-provisioning. Specific contributions of the thesis can be split into four major categories as described next.

1.1.1 Design, Modeling and Analysis of Advanced Network Systems

Today's network systems avoid randomness and diversity to guaranty end-to-end performance, reduce processing overhead related to coding and encryption, and minimize over-provisioning to cut costs

and, thus, limit security and reliability by design. However, all randomness, diversity, data coding and over-provisioning are essential to ensure privacy, anonymity, integrity, and reliability. This thesis designs and models advanced network systems that can provide randomness, diversity, coding, and over-provisioning through random routing, parallel transmission over diverse paths, erasure coding as well as coding and path redundancy, while offering excellent end-to-end performance. To ensure, that the proposed network system meets the requirements of current standards such as IEEE802.3, a developed combinatorial performance analysis allows the evaluation of relevant performance parameters such as delays, i.e., skew and flow completion time, buffer size for deskewing, transmission and coding overhead. The design issues of the advanced network system related to the traffic splitting over parallel lanes and paths, routing and forwarding of the parallelized traffic are addressed and analytically assessed. To this end, two strategies for implementation of a novel random routing are introduced, modeled and analyzed: random circuit switching over pre-reserved not optimized end-to-end paths, and random packet switching without any resource reservation. Finally, a performance of data transmission over the optimized and randomly selected parallel paths with and without coding and with and without over-provisioning is analyzed and compared.

1.1.2 A Combinatorial Reliability Analysis of Advanced Network Services

The failures of software and hardware components in NFV networks can be a result of a misconfiguration, overload or security attack. Thereby, assessing SFC reliability and dimensioning of backup resources as a function of the VNF placement strategy is crucial for optimizing load balancing, resource utilization and throughput in data center networks. Analysis of SFC reliability, however, is not an easy task, as it requires consideration of co-location and sharing

of hardware components, heterogeneity of software and hardware, and complex interdependency between them. To improve the reliability of SFC, there is a need for diversity and over-provisioning such as standard backup protection, the coding, however, can help to simplify and accelerate the failure recovery. This thesis analytically investigates the reliability of two types of advanced network services, which are reliable data transmission and reliable service function chaining. To alleviate the challenges associated with VNF failure detection, VNF migration, and traffic redirection in the case of backup protection, coding is proved to provide service reliability and to simplify the handling of active and backup VNFs and traffic lost. The novel generalized reliability analysis based on combinatorics and probability theory enables evaluation of the SFC reliability as a function of VNF placement strategies in generic inter- and intra-DCN. Here, both failure protection methods with coding and backup are analyzed and compared. The notable feature of the developed analytical approach for evaluation of SFC reliability is the ability to simultaneously consider failures of DC, racks, servers, VMs and path segments as well as component sharing, their heterogeneity, and component interdependency in the case of failures.

1.1.3 Analysis of Security Degree of Advanced Network Systems

Security threats such as eavesdropping and jamming can be performed on the physical layer and IP layers affecting anonymity, privacy, integrity, and reliability of transmitted data. The advanced network systems based on random routing, parallel transmission and erasure coding proposed in this thesis are generally designed to minimize the success of any security attack. However, depending on the location of adversaries, different forwarding and path selection strategies are likely to yield different security performance. Thus, the benefits of parallel transmission, random routing, and coding for

security purposes need to be evaluated. Here, there is a need to analyze the level of provided security in comparison to the standard systems. This thesis provides a novel combinatorial approach for security evaluation considering different types and a different number of adversaries, who can access network links and nodes. As a security measure, a *security degree* is introduced and analyzed as a probability that an attacker can not recover sent data at the first try or disturb communication, which is a function of the number of compromised components in the network. The impact of designed data forwarding, i.e., random circuit switching or random packet switching, on system robustness against different security attacks is studied, whereby the new analytical models take into account different security threats, which can occur simultaneously and in different combinations. The suitability of erasure coding to balance the demands of integrity, reliability and privacy is analyzed for simultaneous jamming and eavesdropping. To reduce the probability of censorship success as a type of jamming attack, a standard Tor network is extended to parallel transmission, random routing, coding, and diverse redundancy, and extensively examined.

1.1.4 Engineering and Benchmarking Anonymous Communication Systems

The main drawback of traditional anonymous networks such as Tor is their overall performance regarding delays and throughput, and limited randomness and diversity resulting in poor anonymity, integrity, and reliability. This thesis investigates the feasibility of anonymous optical networks to ensure both performance and security. Optical networks can be easily extended to provide randomness, path diversity, coding, and over-provisioning, as they can choose among hundreds of wavelengths over a single fiber and erasure coding can be applied at the communication endpoint. The only function that optical networks can not perform today is the layered encryption

all-optically, which requires the design of a fundamentally new all-optical stream cipher (oSC). Here, we design an oSC based on optical XOR (oXOR) and optical LFSR (oLFSR) components with a target to generate a long optical keystream at an optical line rate using a shorter electronic sequence at comparably lower bit rates. As a result, an electronic sub-system, - a cryptographically strong pseudo-random number generator, is cascaded [37] with a cryptographically insecure optical sub-system, i.e., oLFSR. The resulting optical cascade, however, requires new security evaluation techniques due to the bit rate adaptation. Hence, novel security benchmarks were developed based on the information theory and enable the engineering of an anonymous optical network with the required security level.

1.2 Supporting Publications

1.2.1 Journal Articles

1. A. Engelmann and A. Jukan, "A Combinatorial Reliability Analysis of Generic Service Function Chains in Data Center Networks," in *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, under submission.

2. A. Engelmann and A. Jukan, "Toward All-Optical Layered Encryption: A Feasibility Analysis of Optical Stream Cipher," in IEEE Transactions on Information Forensics and Security, vol. 14, no. 10, pp. 2689-2704, Oct. 2019. doi: 10.1109/TIFS.2019.2904793

3. A. Engelmann, W. Bziuk, A. Jukan and M. Médard, "Exploiting Parallelism With Random Linear Network Coding in High-Speed Ethernet Systems," in IEEE/ACM Transactions on Networking, vol. 26, no. 6, pp. 2829-2842, Dec. 2018. doi: 10.1109/TNET.2018.2852562

4. X. Chen, A. Engelmann, A. Jukan and M. Medard, "Linear network coding and parallel transmission increase fault tolerance and optical reach," in IEEE/OSA Journal of Optical Communications and Networking, vol. 9, no. 4, pp. 244-256, April 2017. doi: 10.1364/JOCN.9.000244

5. X. Chen, A. Engelmann, A. Jukan and M. Medard, "Linear Network Coding Reduces Buffering in High-Speed Ethernet Parallel Transmission Systems," in IEEE Communications Letters, vol. 18, no. 4, pp. 636-639, April 2014. doi: 10.1109/LCOMM.2014.013114.132787

1.2.2 Conferences and Workshops

1. A. Engelmann, W. Bziuk and A. Jukan, "Bounding Reliability in Service Function Chaining," 2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 2020, pp. 413-418, doi: 10.23919/MIPRO48935.2020.9245235.

2. A. Engelmann and A. Jukan, "Defying Censorship with Multi-Circuit Tor and Linear Network Coding," 2019 12th CMI Conference on Cybersecurity and Privacy (CMI), Copenhagen, Denmark, 2019, pp. 1-6. doi: 10.1109/CMI48017.2019.8962147

3. A. Engelmann, A. Jukan and R. Pries, "On Coding for Reliable VNF Chaining in DCNs," 2019 15th International Conference on the Design of Reliable Communication Networks (DRCN), Coimbra, Portugal, 2019, pp. 83-90. doi: 10.1109/DRCN.2019.8713668

4. A. Engelmann and A. Jukan, "A Reliability Study of Parallelized VNF Chaining," 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, 2018, pp. 1-6. doi: 10.1109/ICC.2018.8422595

5. A. Engelmann and A. Jukan, "Practical privacy in WDM networks with all-optical layered encryption," 2017 IEEE International Conference on Communications (ICC), Paris, 2017, pp. 1-6. doi: 10.1109/ICC.2017.7996813

6. A. Engelmann and A. Jukan, "Optical Onion Routing," 2017 International Conference on Computing, Networking and Communications (ICNC), Santa Clara, CA, 2017, pp. 323-328. doi: 10.1109/ICCNC.2017.7876148

7. F. Carpio, A. Engelmann and A. Jukan, "DiffFlow: Differentiating Short and Long Flows for Load Balancing in Data Center Networks," 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, 2016, pp. 1-6. doi: 10.1109/GLOCOM.2016.7841733

8. A. Engelmann and A. Jukan, "Balancing the demands of reliability and security with linear network coding in optical networks," 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016, pp. 1-7. doi: 10.1109/ICC.2016.7511590

9. A. Engelmann, S. Zhao and A. Jukan, "Improving Security in Optical Networks with Random Forwarding and Parallel Transmission," 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, 2015, pp. 1-6. doi: 10.1109/GLOCOM.2015.7417629

10. A. Engelmann and A. Jukan, "Away from optimal: performance analysis of multilane ethernet transmission with random path selection in optical networks," 2015 International Conference on Optical Network Design and Modeling (ONDM), Pisa, 2015, pp. 116-121. doi: 10.1109/ONDM.2015.7127284

11. A. Engelmann and A. Jukan, "Serial, parallel or hybrid: Towards a highly reliable transmission in RF/FSO network systems," 2015 IEEE International Conference on Communications (ICC), London, 2015, pp. 6181-6186. doi: 10.1109/ICC.2015.7249308

1.3 Thesis Organization

This thesis is structured in seven chapters. After the introduction, Chapter 2 introduces the concepts of randomness, diversity, coding, and over-provisioning in advanced network systems and analysis for their performance evaluation. Chapter 3 addresses a service reliability issue and presents a novel analytical evaluation of SFC reliability as a function of the VNF placement strategy. A novel combinatorial approach for security evaluation is presented in Chapter 4. Chapter 5 shows a case study of an anonymous optical network focusing on its engineering and new information-theoretical benchmarking. Chapter 6 concludes the thesis. Finally, Chapter 7 provides some mathematical proofs and derivations.

2

Design, Modeling and Analysis of Advanced Network Systems

2.1 Introduction

As randomness, diversity, data coding and over-provisioning are necessary tools to ensure privacy, anonymity, integrity, and reliability, this thesis investigates *advanced network systems* with inherent random routing, parallel transmission, erasure coding as well as diverse redundancy as a possible solution to improve reliability and security.

As parallelism in computing becomes critically important, parallel network systems are following suit. High-speed Ethernet standard IEEE802.3 specifies that the 40/100/400Gb/s Ethernet traffic can be split into data blocks and distributed over multiple parallel lanes (e.g., 10 lanes x 10Gb/s, for 100Gb/s), referred to as Multi-Lane Distribution (MLD) [2]. Thus, high-speed Ethernet traffic can be distributed to multiple low-speed optoelectronic interfaces addressing the need of today's applications for a large aggregated bandwidth. Each lane can be mapped onto parallel output interfaces and optical channels, i.e., optical paths, for parallel transmission in Optical Transport Networks (OTNs). Transmission over multiple parallel paths in optical networks is a well-studied subject, known, however, for a strict constraint on optimality [15, 16]. The optimization goal

is usually to minimize the inter-lane skew, i.e., the time difference between receiving the data on the first and the last lane. The skew of data blocks, which occurs not only due to path diversity but also on the packetization process in the end-system, is hard to assess and guarantee. Thus, in addition to paths optimization, there is a need for the so-called *de-skewing* via buffering at the receiver [2]. However, the High-speed Ethernet standard IEEE802.3 defines, in fact, the maximum skew of only 180ns to avoid buffering of unrecoverable Ethernet frames. In Data Center Networks (DCNs), Equal-Cost Multi-Path (ECMP) is used as the de-facto routing protocol, which can split the traffic into multiple sub-flows and sent them over multiple paths following flow- or packet-based routing. An extension of ECMP to the parallel transmission was proposed in [17, 18, 19], whereby the parallel sub-flows are uniformly distributed over the network to improve the throughput, server utilization, and load balancing. Since DCN topology offers paths with an equal length between any source-destination pair, the only issue of parallel transmission in DCN is to meet a Flow Completion Time (FCT), i.e., an end-to-end path and flow transmission delay.

For the first time, a random routing in the packet-switched network was investigated as a possible routing strategy in military communication networks, whereby each message path represents a random walk [38]. For circuit-switched networks, a random routing was first analyzed in [39] as a decentralized routing. Although it was shown that the random routing is extremely reliable in presence of any adversary but highly inefficient concerning end-to-end delay, i.e., impractical, the research community has been still investigating a possible design, performance, and practicality of random routing. That is not least because of its decentralized nature, low control and management overhead and independence of network structure. For modern networks such as High-Speed Ethernet as well as network topologies such as in DCNs, the random routing in combination with

the parallel transmission is less studied and needs to be proved to provide all its advantages and required delay performance.

To address data integrity and reliability, an erasure code can be applied to the parallelized traffic. The coding redundancy can be added and sent over additional paths, i.e., path redundancy, to protect the communication system against both data and path losses as shown in [24], where coding redundancy allows performing erasure correction at receiver. Moreover, it was shown in [40, 41, 42] that coding can outperform optimal routing in a single unicast setting. Currently, many MDS-based erasure codes (Maximum Distance Separable Codes) have been proposed, such as Reed-Solomon [20] or regenerating codes [21] based on Network Coding [22] or a random Linear Network Code (LNC) [23], which offer good trade-offs in terms of redundancy, reliability, and repair bandwidth.

This chapter designs advanced network systems that can exploit random routing, parallel data transmission over diverse paths, erasure coding as well as coding and path redundancy. New solutions are provided to address the challenges of parallel transmission from the perspectives of design, modeling and analysis, for both routings over the packet-switched and circuit-switched networks. To prove the suitability of designed parallel transmission, we provide novel performance analysis based on combinatorics and investigate whether routing optimality is necessary for parallel network systems and whether coding can relax requirements on paths optimality. The new performance analysis allows assessing delays, the receiver buffer and the overall realization overhead of parallel transmission in different settings. The comprehensive analysis and comparison of parallel transmission over the optimized and randomly selected parallel paths with and without coding are provided with a focus on delays, e.g., skew and FCT, and required deskew buffer.

2.2 Supporting Publications

1. A. Engelmann, W. Bziuk, A. Jukan and M. Médard, "Exploiting Parallelism With Random Linear Network Coding in High-Speed Ethernet Systems," in IEEE/ACM Transactions on Networking, vol. 26, no. 6, pp. 2829-2842, Dec. 2018. doi: 10.1109/TNET.2018.2852562

2. X. Chen, A. Engelmann, A. Jukan and M. Medard, "Linear network coding and parallel transmission increase fault tolerance and optical reach," in IEEE/OSA Journal of Optical Communications and Networking, vol. 9, no. 4, pp. 244-256, April 2017. doi: 10.1364/JOCN.9.000244

3. X. Chen, A. Engelmann, A. Jukan and M. Medard, "Linear Network Coding Reduces Buffering in High-Speed Ethernet Parallel Transmission Systems," in IEEE Communications Letters, vol. 18, no. 4, pp. 636-639, April 2014. doi: 10.1109/LCOMM.2014.013114.132787

4. A. Engelmann and A. Jukan, "Away from optimal: performance analysis of multilane ethernet transmission with random path selection in optical networks," 2015 International Conference on Optical Network Design and Modeling (ONDM), Pisa, 2015, pp. 116-121. doi: 10.1109/ONDM.2015.7127284

5. F. Carpio, A. Engelmann and A. Jukan, "DiffFlow: Differentiating Short and Long Flows for Load Balancing in Data Center Networks," 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, 2016, pp. 1-6. doi: 10.1109/GLOCOM.2016.7841733

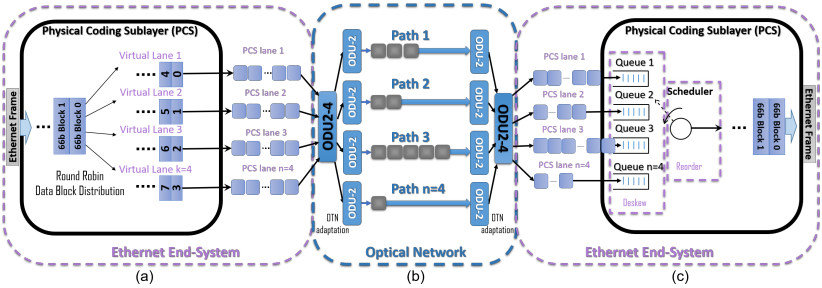


Figure 2.1: Multi-lane Ethernet-over-optical network system [1]. (a) Traffic distribution in sender; (b) parallel transmission over the optical network and skew formation; (c) deskew buffers and reordering from [2].

2.3 Modeling and analysis of Parallel Transmission

The concept of the parallel transmission can be considered on an example of the end-system according to the IEEE802.3 standard [2]. Fig. 2.1 shows a typical multi-lane 40Gb/s Ethernet-over-optical network end-system. In the sender (Fig. 2.1(a)), a high-speed stream of serial Ethernet frames is split into data blocks of 64b, encoded with 64b/66b line code in Physical Coding Sublayer (PCS), and distributed over, for instance, $k = 4$ virtual Ethernet lanes. For identification of the lane ordering at the receiver, specific alignment markers are inserted in each lane after $m_L = 16383$ data blocks. Finally, the parallel lanes are mapped to optical paths of the underlying optical network. This mapping can follow two different routing mechanisms that are optical circuit switching and optical packet switching.

As shown in Fig. 2.1(b), in the circuit-switched networks such as Wavelength Division Multiplexing, each 10GE Ethernet lane can be mapped to four optical data units (ODU), here of type 2. The ODU2e method enables the transparent 10GE mapping using an over-clocking approach, whereas Generic Framing Procedure (GFP)

principles are applied. The ODU signals are modulated on four optical carriers and transmitted over $n = k = 4$ optical paths. In general, the number of virtual Ethernet lanes k and the allocated optical paths n do not need to be equal. In our example, however, OTN is assumed to map data streams from Ethernet lanes into $n \geq k$ ODU2-n or ODU2e-n containers.

A simplified architecture of the receiver, also according to IEEE802.3, is shown in Fig. 2.1(c). Here, the PCS layer processes the 66b data blocks received to retrieve the original Ethernet frame, which requires multiple processing entities, including lane block synchronization, lane deskew and reorder, alignment removal, etc. (not all shown here). To illustrate the skew and deskewing process, let us assume that paths 3 and 4 are the shortest and the longest path in terms of the end-to-end delay, respectively. The time difference between arrivals of the first data block from path 3 and the first data block from path 4 is referred to as skew. For compensation of the resulting inter-lane skew, the data blocks from path 3 must be buffered in the receiver until data blocks from longer paths arrive, i.e. from paths 1, 2, and, finally, from path 4. For skew compensation, the receiver implements the so-called *deskewing* and *reordering*. The deskew function of the PCS is implemented with the parallel input FIFO queues, that store data blocks until the alignment markers of all lanes are received and synchronized. That allows the scheduler to start the lane identification, alignment removal and reordering to form the original serial data stream.

Another possible solution to map the parallel lanes in Ethernet-over-optical network end-systems would be optical packet switching (OPS). In contrast to the circuit-switched networks, the optical paths between a certain source-destination pair do not need to be reserved, whereby the forwarding is destination based and hop-by-hop. The optical packets are forwarded to a chosen free port associated with a certain destination at every optical packet switch. When a

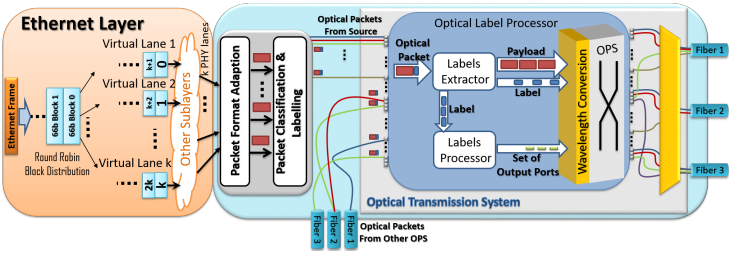


Figure 2.2: A possible realization of an optical packet switching [3].

source sends optical packets in parallel, the end-to-end routes and the number of utilized paths can dynamically change during traversing the network. As a result, the packets do not only experience skew, but also can arrive at any available ports at the receiver. The design of the OPS system is potentially more technologically challenging and deserves discussion. Fig 2.2 shows how the potential multilane Ethernet-over-OPS system can be realized. Here, multiple data blocks from PHY Ethernet lanes can be formed into an optical packet of a defined length (per each lane). This is a function of Packet Format Adaptation. In the functional block of Packet Classification and Labeling, the optical label is defined according to the destination. At every optical packet switch, all-optical packets, possibly from multiple sources, arrive at the input ports and need to be forwarded. The function of forwarding can be implemented similarly to [43], whereby each optical packet passes through the optical Label Extractor, which extracts the labels of each optical packet and sends it to the Label Processor.

The Label Processor analyzes the label and determines all candidate output ports to a destination and all available wavelengths considering the wavelength continuity constraint. The forwarding tables can be configured via a control plane. Per our model, the Label Processor has information about all possible next hops towards the destination and selects the available ports and wavelengths entirely

randomly. If the input wavelength differs from the output wavelength, wavelength conversion is required before switching or else the packet needs to be dropped or buffered [44]. After label extraction, the optical payload and extracted label are passed to the optical switch, where the extracted label is added again to the head of the optical payload before switching. It is important to note that depending on the performance of optical switch components, i.e., Label Extractor and Label Processor, and their implementation, the optical payload may be delayed in the process, e.g., optically buffered. Thus, it is necessary to synchronize payload with labels.

In generality, the traffic parallelization can be implemented on TCP Layer as well, whereby routing and forwarding on IP Layer can follow circuit switching or packet switching principles. For instance, the mapping of parallel TCP flows into IP tunnels is possible using Equal Cost Multipath Protocol (ECMP), which can be packet or flow-based corresponding to circuit or packet switching, respectively.

2.3.1 Network Model

For modeling and performance analysis of parallel transmission between a certain source-destination pair, we utilize notation summarized in Sec. 2.7. The network is assumed to represent a directed and acyclic graph $G(V, E) = G$, where V and E are vertex set and edge set, respectively. The source node is denoted as $s \in V$, while the destination node is labeled as $d \in V$. A distinction is made between incoming and outgoing links of an arbitrary node $v \in V$, denoted as a set $\mathcal{E}_{in}(v)$ and $\mathcal{E}_{out}(v)$, respectively. Each node has $|\mathcal{E}_{in}(v)| = y_{in}$ incoming and $|\mathcal{E}_{out}(v)| = y_{out}$ outgoing links. The traffic, k parallel data flows, is modeled as a binary sequence, whereby each flow is decomposed into *packets or data blocks* of the same length. The *time unit (tu)* in G is a discrete time based on the link capacity and analyzed as a transmission delay of one packet.

Without loss of generality, we assume that the network provides $\mathfrak{N}$

available parallel paths between s and d suitable for the certain routing and forwarding method to realize the parallel traffic transmission over $G(V, E)$. Then, that $\mathfrak{N}$ available parallel paths $\mathcal{P}_l(s,d) \equiv \mathcal{P}_l$, $l = 1, ..., \mathfrak{N}$, are collected in a set $G_{\mathcal{P}} = \{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_l, ..., \mathcal{P}_{\mathfrak{N}}\}$ and are sorted in the ascending order so that the increasing index l of each path $\mathcal{P}_l$ corresponds to an increasing path delay d_l, i.e., $d_1 \leq d_2 ... \leq d_l, ..., \leq d_{\mathfrak{N}}$, which are arranged in a vector of length $\mathfrak{N}$, $\vec{d} = (d_1, d_2, ..., d_{\mathfrak{N}})$. Some of the available paths can have the same path delay, i.e., $d_{l-1} = d_l = d_{l+1}$. To simplify the further analysis, we assume integer values for the path delays, i.e. $\lfloor d_l \rfloor = d_l$, which can be realized without loss of accuracy by choosing a sufficiently small value of the *time unit (tu)*.

Generally, any path $\mathcal{P}_l$ is a chain of $\mathfrak{y}_l$ links noted as $\rho_{lh} \in \mathcal{P}_l$, $1 \leq h \leq \mathfrak{y}_l$, and of $\mathfrak{y}_l - 1$ intermediate nodes $v_{lq} \in \mathcal{P}_l$, $1 \leq q \leq \mathfrak{y}_l - 1$. The node-to-node delay between two directly connected nodes on the certain path $\mathcal{P}_l$ is denoted as ε_{lh}. Thus, if there are no additional delays or blocking in the intermediate nodes, the end-to-end path delay of any path $\mathcal{P}_l$ between source s and destination d is

$$d_l = \sum_{\substack{h=1, \\ \rho_{lh} \in \mathcal{P}_l}}^{\mathfrak{y}_l} \varepsilon_{lh}. \tag{2.1}$$

When n parallel paths among $\mathfrak{N}$ available paths are selected from a subset $G_{\mathcal{P}}$, there is a set $\mathcal{M}$ of all possible path subsets $\mathcal{M}_n(\alpha)$, i.e., there are $|\mathcal{M}| = a_{\mathcal{M}} = C_{\mathfrak{N},n}$[1] possible path combinations, where each combination is collected in a subset $\mathcal{M}_n(\alpha) \in \mathcal{M}$, $1 \leq \alpha \leq a_{\mathcal{M}}$. However, all paths in each subset $\mathcal{M}_n(\alpha)$ are sorted so that their corresponding path delays appear in ascending order. This requires to derive an index mapping l_m, which maps an index m, $m = 1, 2, ..., n$, used to specify a path $\mathcal{P}_{l_m}(\alpha)$ out of the subset $\mathcal{M}_n(\alpha) = \{\mathcal{P}_{l_1}(\alpha), ..., \mathcal{P}_{l_m}(\alpha), ..., \mathcal{P}_{l_n}(\alpha)\}$ to a path $\mathcal{P}_l \in G_{\mathcal{P}}$, $l = 1, 2, ..., \mathfrak{N}$. This also maps the delay $d_{l_m}(\alpha)$ of path $\mathcal{P}_{l_m}(\alpha)$ to the corresponding

[1] We note the binomial coefficient $\binom{i}{j}$ as $C_{i,j}$, whereby $C_{i,j} = 0$ for $j > i$.

component d_l of the delay vector $\vec{d}$ so, that the increasing index m of each path $\mathcal{P}_{l_m}(\alpha)$ corresponds to an increasing path delay $d_{l_m}(\alpha)$.

To define such mapping let us introduce $\mathcal{N} = \{1, 2, ..., \mathfrak{N}\}$ and $\mathcal{N}_n = \{\Lambda_n \subset \mathcal{N} : |\Lambda_n| = n\}$, i.e. the set of all subsets of $\mathcal{N}$ with cardinality n. Let us use $\Lambda_n(\alpha)$, $\alpha = 1, 2, ..., a_{\mathcal{M}}$, to index each of these subsets. Thus based on paths $P_l \in G_{\mathcal{P}}$, the set $\mathcal{M}_n(\alpha)$ is given as $\mathcal{M}_n(\alpha) = \{\mathcal{P}_l : l \in \Lambda_n(\alpha)\}$ and the final mapping between path $\mathcal{P}_{l_m}(\alpha) \in \mathcal{M}_n(\alpha)$ and $\mathcal{P}_l \in G_{\mathcal{P}}$ is defined by the index function

$$\delta_\alpha(l, m) = \begin{cases} 1, & \text{if the } m^{th} \text{ element of } \Lambda_n(\alpha) \text{ is } l, \\ 0, & \text{otherwise,} \end{cases} \tag{2.2}$$

where $l \in \mathcal{N}$ and $m = 1, 2, ..., n$.

Furthermore, Eq.(2.2) enables the mapping of paths by the relation $\mathcal{P}_{l_m}(\alpha) \equiv \sum_i \mathcal{P}_i \delta_\alpha(i, m)$ and has the property $\sum_l \delta_\alpha(l, m) = 1$. For example, when $\mathfrak{N} = 6$ and $n = 3$ parallel paths are chosen from the set $G_{\mathcal{P}}$, e.g., $\mathcal{P}_3, \mathcal{P}_4$ and $\mathcal{P}_5$, the sorting due to increasing delays is defined by the mapping to the selected subset shown by the equivalence $\mathcal{M}_n(\alpha) = \{\mathcal{P}_{3_1}(\alpha), \mathcal{P}_{4_2}(\alpha), \mathcal{P}_{5_3}(\alpha)\} \equiv \{\mathcal{P}_3, \mathcal{P}_4, \mathcal{P}_5\}$.

Modeling of Circuit Switching

In the circuit-switched network, F different link disjoint paths exist and can be established between the given s and d. At the same time, some of the links belonging to the set of F paths can be generally considered for path requests between other source-destination node pairs. Thus, depending on the traffic load offered to the network, at any connection request time, a $s-d$ pair can utilize at most $N \leq F$, $N \equiv \mathfrak{N}$, disjoint paths, whereas it cannot use the remaining $F - N$ paths due to some links being used by other connection requests. Let us, in fact, assume that any set of at least N out of F parallel paths is available for a given connection request, whereby $n \leq N$ paths are then selected for parallel transmission. Since all path links

are pre-reserved for transmission, the end-to-end path delay d_l of any path $\mathcal{P}_l$ is known and determined by Eq. (2.1).

To derive the likelihood that network can provide $N \geq n$ available paths, let us denote the setup probability of an arbitrary path $\mathcal{P}_l$ as P_{setup}. However, we do not consider how various traffic load pattern impact the path setup probability P_{setup}. The model for blocking probability of the connection request $B(n)$ can be approximated by assuming that the network load is distributed so that each out of F possible paths can be set up with an equal probability, P_{setup}. As a result, the probability that a path $\mathcal{P}(s,d)$ between s and d cannot be set up, and is blocked, is defined as

$$P_{\text{B}} = 1 - P_{\text{setup}}. \tag{2.3}$$

Thus, the probability of N available parallel paths out of F existing paths would follow the Binomial distribution, i.e.,

$$Pr(N = j) = \binom{F}{j} (1 - P_{\text{B}})^j P_{\text{B}}^{F-j}. \tag{2.4}$$

Finally, the transmission request is blocked with probability $B(n)$, when the number of available paths N is lower than the number of outgoing interfaces n, i.e., $N < n$.

$$B(n) = \sum_{j=0}^{n-1} Pr(N = j). \tag{2.5}$$

The mean number of available paths is determined as follows

$$\bar{N} = \sum_{j=0}^{F} j \cdot Pr(N = j) = F \cdot P_{\text{setup}}. \tag{2.6}$$

On the other hand, the mean number of paths available $E\{N|N \geq n\}$, which ensures the successful parallel transmission and is relevant for further analysis, is given by

$$E\{N|N{\geq}n\}{=}\sum_{j=0}^{F} j \cdot Pr(N{=}j|N{\geq}n){=}\frac{\sum_{j=n}^{F} j{\cdot}Pr(N{=}j)}{1 - B(n)}. \tag{2.7}$$

In the optical circuit-switched networks, the path model needs to take into consideration that each fiber-link provides many different wavelengths, where the wavelength is used as a synonym for the link. Let F_p be the number of different disjoint fiber paths between an arbitrary pair of nodes, where each fiber path offers W_o different optical (wavelength) paths. Thus, a total number of $F = F_p \times W_o$ optical paths (synonym for path) exists for the selected $s - d$ pair.

Modeling of Packet Switching

In the packet-switched network, we intend a directed and acyclic network graph $G(V, E)$ not to represent the full network, but only to contain those network nodes with available output interfaces associated with a certain destination, as selected from the set of all nodes in the network at the moment of transmission request. That results in $\Psi \geq N$, $\Psi \equiv \mathfrak{N}$, available and not necessarily disjoint paths, whereby two or more paths can compete for the same outgoing link. The network model does not consider different traffic loads and assumes that any outgoing link can be available with equal probability. That requires, however, that all outgoing links have the same link capacity. Here, a source node s can send $n = k$ packets in parallel using $n \leq |\mathcal{E}_{out}(s)|$ outgoing links, while at most $\Psi \geq n$ available paths can be simultaneously utilized over the network toward the destination d by choosing the next hop based on available output interfaces in each network node. Generally, any path $\mathcal{P}_l$, $1 \leq l \leq \Psi$ consists of $\mathfrak{y}_l$ links ρ_{lh}, $1 \leq h \leq \mathfrak{y}_l$, where the probability to utilize link ρ_{lh} is defined as ϕ_{lh}. Then, the occurrence probability of any path $\mathcal{P}_l$ is determined as follows

$$P_l = \prod_{h=1}^{\mathfrak{y}_l} \phi_{lh}. \tag{2.8}$$

Since the packets parallelized in source s into k flows can be distributed over n, $k \leq n \leq \Psi$, available paths, the packet switching

principle allows sending the packets in parallel over multiple network paths. In the network nodes, the packets do not necessarily compete for the same outgoing link at the same time. For instance, a packet can be sent over any available and directed outgoing link to avoid packet dropping or buffering. However, we next combinatorically analyze the occurrence probability of paths that lead to congestion.

On each intermediate node $v \in V$, the number of incoming links can be larger than the number of available outgoing links, i.e. $y_{in} > y_{out}$. When a forwarding node v receives ω packets over $y_{in}(v)$ incoming links simultaneously, i.e., $y_{out}(v) < \omega \leq y_{in}(v)$, then, $\omega - y_{out}(v) \geq 0$ packets can not be forwarded and, thus, are dropped or buffered with probability $P_B(v)$. In this case, we refer to the node v as a congested node, whereby the ratio of the congested nodes is determined as $\mathfrak{G} = \frac{|V'|}{|V|}$, when network G consists of $|V|$ nodes and $|V'|$ out of $|V|$ nodes are congested, i.e., $V' \in V$. Generally, the source s and an arbitrary congested node v, $y_{out}(v) < y_{in}(v)$, are connected by maximal Ψ_{sv} paths, which can deliver at most $\omega = y_{in}(v)$ packets per tu arriving on $y_{in}(v)$ out of $|\Psi_{sv}|$ paths simultaneously. Thus, there are $a_{\mathcal{M}'} = \sum_{i=1}^{y_{in}(v)-y_{out}(v)} \binom{\Psi_{sv}}{y_{out}(v)+i}$ paths combinations, which can lead to blocking of $\omega - y_{out}(v)$ packets in the node v. In addition, since some out of Ψ_{sv} paths can be partly disjoint, i.e., contain common links such as $\rho_{lh} = \rho_{(l+1)(h+1)}$, from different paths, $\rho_{lh} \in \mathcal{P}_l(s,v)$ and $\rho_{(l+1)(h+1)} \in \mathcal{P}_{(l+1)}(s,v)$, but can be used only by a one path at time, we apply the rule, which constraints using a common links by multiple paths [45] as follows $\phi_{lh} \cdot \phi_{(l+1)(h+1)} = \phi_j$, where $\phi_{lh} = \phi_{(l+1)(h+1)} = \phi_j$ is a probability to utilize a link $e_j \in E$.

Let set $\mathcal{M}'_{\omega}(\alpha)$ be a set of the ω selected paths out of Ψ_{sv} available paths between any pair s and v in the paths combination α and a set $\bar{\mathcal{M}}'_{\Psi_{sv}-\omega}(\alpha) = \{\mathcal{P}_{l_m}(s,v) | \mathcal{P}_{l_m}(s,v) \notin \mathcal{M}'_{\omega}(\alpha)\}$ a set of available paths, which are not utilized for transmission in the paths combination α. The probability $\hat{P}(\alpha, \omega, \Psi_{sv})$, that all paths from set $\mathcal{M}'_{\omega}(\alpha)$

and not from set $\bar{\mathcal{M}}'_{\Psi_{sv}-\omega}(\alpha)$ are selected for transmission is defined by Eq. (2.9), where $1 \leq \omega \leq y_{in}(v)$ and $\Psi \equiv \Psi_{sv}$. Without loss of generality, the probability that ω packets are delivered by ω out of Ψ available paths from set $\mathcal{M}_{\omega}(\alpha)$ can be determined as follows

$$\hat{P}(\alpha, \omega, \Psi) = \prod_{\substack{m=1,\\ P_{l_m}(\alpha) \in \mathcal{M}_{\omega}(\alpha)}}^{\omega} P_{l_m}(\alpha) \prod_{\substack{m=1,\\ P_{l_m}(\alpha) \in \bar{\mathcal{M}}_{\Psi-\omega}(\alpha)}}^{\Psi-\omega} (1 - P_{l_m}(\alpha)), \quad (2.9)$$

where the path probability $P_{l_m}(\alpha)$ of a path $\mathcal{P}_l$ is defined by Eq. (2.8), i.e., $P_{l_m}(\alpha) \equiv P_l$, and $\mathcal{M}_{\omega}(\alpha)$ is a set of ω paths out of Ψ available paths between source s and destination d. When $y_{in}(v_{lq}) > y_{out}(v_{lq})$, the probability of paths competition for the same outgoing link in an arbitrary node v can be determined as follows

$$P_B(v) = \sum_{\omega=y_{out}(v)+1}^{y_{in}(v)} \sum_{\alpha=1}^{a_{\mathcal{M}'}} \hat{P}(\alpha, \omega, \Psi_{sv}). \quad (2.10)$$

When more than one packet claims the same outgoing link at the same time in the intermediate node v_{lq} on the path $\mathcal{P}_l$, and, thus, needs to be buffered, each packet can be delayed with probability $P_B(v_{lq})$ (Eq. (2.10)) for ϱ_q *time units*. When the forwarding in the intermediate nodes follows the FIFO rules, the resulting mean end-to-end delay d_l of path $\mathcal{P}_l(s,d) \equiv \mathcal{P}_l$ is defined as follows

$$d_l = \sum_{\substack{h=1,\\ \rho_{lh} \in \mathcal{P}_l}}^{\mathfrak{y}_l} \varepsilon_{lh} + \sum_{\substack{q=1,\\ v_{lq} \in \mathcal{P}_l}}^{\mathfrak{y}_l - 1} P_B(v_{lq}) \cdot \varrho_q. \quad (2.11)$$

In optical packet-switched networks, any optical node can support u_{in} and u_{out} input and output interfaces, respectively. Since each optical fiber can provide different wavelengths, each input and output interface can provides w_{in} and w_{out} available wavelengths, respectively. Thus, each input or output interface provide w_{in} or w_{out} parallel links between two neighbor nodes. That results in $|\mathcal{E}_{in}(v)| = u_{in} \cdot w_{in}$ available incoming links and $|\mathcal{E}_{out}(v)| = u_{out} \cdot w_{out}$

available outgoing links at any arbitrary optical node v. When the priority of any optical node v with $u_{out}(v)$ available interfaces is to avoid wavelength conversion, i.e., to forward an optical packets on the output wavelength λ_{out} corresponding to the input wavelength, λ_{in}, $\lambda_{out} = \lambda_{in}$, the number of links is limited to $u_{out}(v)$. That results in a network model equivalent to the general model above, i.e., $u_{out}(v) \equiv y_{out}(v)$, where a path delay includes a queuing delay and can be calculated with Eq. (2.11). In contrast, when the wavelength conversion is applied to avoid the packet blocking and the optical buffering the number of incoming and outgoing links is defined as $y_{in}(v) = u_{in} \cdot w_{in}$ and $y_{out}(v) = u_{out} \cdot w_{out}$, respectively.

2.3.2 Modeling of Routing Strategies

To direct parallelized traffic flows through the circuit-switched and packet-switched networks, we distinguish between two path selection methods: optimal and random. Next, we analyze and compare both methods, i.e., the parallel transmission over the optimized parallel paths and the randomly selected parallel paths, applied in the circuit and packet-switched networks.

Optimized Parallel Transmission

Paths optimization can be implemented in both types of considered network, the circuit-switched networks and the packet-switched networks, which we refer, in this case, to as an optimal routed parallel transmission and an optimal packet forwarding, respectively. In the optimal routed parallel transmission, we refer to the parallel paths as optimal, when these n out of N available parallel disjoint paths were selected to yield the minimal skew. Similarly, the optimal packet switching can select and pre-configure those input and output interfaces in any forwarding node, which result in n out of Ψ parallel paths, where $k \leq n \leq \Psi$, which can yield the minimal skew. For that

reason, for any parallel transmission, each intermediate forwarding node v can provide at most $y_{out}(v) \geq y_{in}(v)$ pre-reserved outgoing links to avoid congestion and additional delays. However, when y_{in} selected optimal paths share the same intermediate node v, there is no fixed mapping between input and output interfaces, whereby any packet is forwarded by the node v over one available out of y_{out} pre-configured outgoing links.

Thus, for both, i.e., the optimal routed parallel transmission and the optimal packet switching, the optimal path set will minimize the skew between selected paths and, thus, the buffer size. When the source selects n optimized parallel paths in the network, there is a fixed optimal path pattern $\mathcal{M}_n(\alpha_{\text{opt}})$ with delays $d_{l_1}(\alpha_{\text{opt}}) \leq d_{l_2}(\alpha_{\text{opt}}) \leq ... \leq d_{l_n}(\alpha_{\text{opt}})$. However, due to the random selection of outgoing links in any intermediate node in optical switched networks, a more complex packet reordering mechanism can be required at the destination. The optimized parallel transmission is commonly used by standard end-systems in today's high-performance networks such as Ethernet/OTN networks and for data center networks applying equal cost multipath (ECMP) routing.

Random Circuit Switching

Without optimization, the source selects any n available paths randomly and with the same probability among all N available paths. As a result, each path combination collected in a subset $\mathcal{M}_n(\alpha) \in \mathcal{M}$, $1 \leq \alpha \leq a_{\mathcal{M}}$ appears with same probability $P(\mathcal{M}_n(\alpha)) = 1/a_{\mathcal{M}}$. Let us define the likelihood that a specific subset $\mathcal{M}_n(\alpha)$ is chosen by indicating its dependence on the number n of parallel paths. Since only one out of $a_{\mathcal{M}}$ path combinations is randomly selected for transmission, all path sets and all paths of a set occur with the same probability. Let us denote the probability of an arbitrary path

combination as $P(\mathcal{M}_n(\alpha)) = \frac{1}{a_{\mathcal{M}}} \equiv P'(\alpha, n)$, whereby

$$P'(\alpha, n) = \prod_{m=1}^{n} P_{l_m}(\alpha) = (P_{\mathcal{M}})^n = \frac{1}{C_{N,n}}, \tag{2.12}$$

where $P_{\mathcal{M}}$ is the probability that an arbitrary path $\mathcal{P}_l \in G_{\mathcal{P}}$ is selected for transmission and collected in $\mathcal{M}_n(\alpha)$.

Random Packet Switching

In the packet-switched networks with random settings, paths or interfaces in forwarding nodes do not need to be reserved, whereby the forwarding can be hop-by-hop, following randomly chosen free interfaces at every forwarding node. When a source sends packets in parallel, these packets do not only experience skew, but also can arrive at any available ports at the receiver. For instance, when a source uses n outgoing links for the first hop to the next nodes, the destination may receive the packets on all its incoming links. Since packets are sent over a randomly changing sequence of links, they can be spread over the packet-switched network so that more than n packets arrive at the destination simultaneously.

Here, the occurrence probability of random paths combination is defined by Eq. (2.9), whereby the paths can interfere and overlap. Since any node v in the packet switched network randomly decides about outgoing link $e_j \in \mathcal{E}_{out}(v)$ for transmission, all outgoing links e_j have the same probability ϕ_{e_j} to be selected, while the probability of a link $e_j \in \mathcal{E}_{out}(s)$ at the source node is defined as $\phi_{e_j} = \sqrt[n]{\frac{1}{C_{|\mathcal{E}_{out}(s)|,n}}}$. When any node v has $y_{out}(v)$ available links, the probability ϕ_{e_j} to select link e_j for transmission is defined as

$$\phi_{e_j} = \frac{1}{y_{out}(v)}. \tag{2.13}$$

2.3.3 Combinatorial Delay Analysis

This subsection provides a derivation of the expected value of the flow completion time (FCT) and the skew in arbitrary networks, the analysis of occurrence probability of maximum skew, analysis of the impact of path failures and packet loss on the skew.

The FCT of a data flow is defined as the difference between two time stamps, the first time stamp holds the time when the first packet of a flow leaves a source, and the second time stamp holds the time when the last packet of the same flow arrives at the destination. If there is no queuing delay in the forwarding nodes, the FCT is the best achievable ("ideal"), where only the maximal end-to-end path delay $d_{\max}$ and the transmission time of the data blocks needs to be considered. Thus, within selected set $\mathcal{M}_n(\alpha)$, $d_{\max}(\alpha) = d_{l_n}(\alpha)$ while the FCT can be determined as follows

$$\mathcal{T}'(\alpha) = \frac{ML_M}{nC} + d_{l_n}(\alpha), \tag{2.14}$$

where M and L_M is the number of data blocks in the source data and bits in each data block, respectively, while C is a transmission bit rate of any path $\mathcal{P}_{l_n}(\alpha)$ chosen for transmission.

The skew is typically defined as difference in delays $d_{\max}$ and $d_{\min}$ between the longest and the shortest paths, respectively. Thus, within chosen set $\mathcal{M}_n(\alpha)$, these delays are determined as $d_{\max}(\alpha) = d_{l_n}(\alpha)$ and $d_{\min}(\alpha) = d_{l_1}(\alpha)$, respectively, resulting in the skew:

$$\tau(\alpha) = d_{\max}(\alpha) - d_{\min}(\alpha). \tag{2.15}$$

Let us define a *path-specific skew* as a delay difference between the longest and an arbitrary path in a path set $\mathcal{M}_n(\alpha)$:

$$\tau_m(\alpha) = d_{l_n}(\alpha) - d_{l_m}(\alpha), \tag{2.16}$$

where $m = 1, 2, ..., n$ and $\tau_1(\alpha) = \tau(\alpha)$ refers to the largest skew within the path set (see Eq. (2.15)). To evaluate the receiver buffer size

required for de-skewing, the cumulative skew between all paths selected is an important parameter and given as a function of utilized parallel paths n and path set selected, i.e.,

$$\tau(n, \alpha) = \sum_{m=1}^{n} \tau_m(\alpha), \tag{2.17}$$

where $\tau_m(\alpha)$ is defined by Eq. (2.16).

As we consider networks with a large number of path sets available, the skews depend on the path set chosen, i.e., on α, $\alpha = 1, ..., a_{\mathcal{M}}$, and, thus, on the path selection method.

Optimized Parallel Transmission

For the optimal routed parallel transmission and the optimal packet switching, all n utilized paths are selected to minimize the skew between chosen paths and, thus, the buffer size. Thus, the largest FCT and skew within the optimal path set is denoted as $\mathcal{T}'(\alpha_{\text{opt}})$ and $\tau(\alpha_{\text{opt}})$, respectively. With Eq. (2.17), the optimal path set is given by $\mathcal{M}_n(\alpha_{\text{opt}}) = \mathcal{M}_n(\alpha)$ for $n = k$, when $min_\alpha\{\tau(k, \alpha)\}$. Then, with Eq. (2.14), FCT is determined as follows

$$\mathcal{T}'(\alpha_{\text{opt}}) = \frac{ML_M}{nC} + d_{l_n}(\alpha_{\text{opt}}) = \mathcal{T}'. \tag{2.18}$$

The skew is known, fixed and determined with Eq. (2.15) as

$$\tau(\alpha_{\text{opt}}) = d_{\max}(\alpha_{\text{opt}}) - d_{\min}(\alpha_{\text{opt}}) = \tau, \tag{2.19}$$

where $\mathcal{T}'$ and τ are known for the certain optimal path set $\mathcal{M}_n(\alpha_{\text{opt}})$.

Random Circuit Switching

Here, we need to analyze the expected values of flow completion time and skew over all $a_{\mathcal{M}}$ possible path combinations collected in $\mathcal{M}$. We next derive the expected value of both delays given a number of available paths N in an arbitrary network, where a set of n parallel paths is chosen randomly for transmission.

Based on Eq. (2.15), the expected value of skew $\bar{\tau} = E_{\mathcal{M}}\{\tau(\alpha)\} = E_{\mathcal{M}}\{d_{\max}(\alpha) - d_{\min}(\alpha)\}$ takes into account all possible paths combinations, as indexed by $\mathcal{M}$, i.e.,

$$\bar{\tau} = E_{\mathcal{M}}\{\tau(\alpha)\} = E\{d_{\max}(\alpha)\} - E\{d_{\min}(\alpha)\}, \tag{2.20}$$

where we use $E_{\mathcal{M}}\{\cdot\} = E\{\cdot\}$ to simplify notation. The expected value $\bar{\tau}$ can be derived knowing the expected delay $E\{d_m(\alpha)\}$ of the m^{th} path $\mathcal{P}_{l_m}(\alpha)$ over all path sets $\mathcal{M}_n(\alpha)$ randomly chosen with probability $P(\mathcal{M}_n(\alpha)) = 1/a_{\mathcal{M}}$. Using the mapping $d_{j_m}(\alpha) = \sum_l d_l \delta_\alpha(l,m)$ as specified by Eq. (2.2), the expected delay of m^{th} path can be derived as follows

$$E\{d_m(\alpha)\} = \sum_{\alpha} \sum_{l} d_l \delta_\alpha(l,m) P(\mathcal{M}_n(\alpha)) = \sum_{l} d_l p_l(m), \tag{2.21}$$

where $p_l(m) = 1/a_{\mathcal{M}} \sum_\alpha \delta_\alpha(l,m)$ is the probability, that the l^{th} path $\mathcal{P}_l$ with delay d_l from $\vec{d}$ is selected as m^{th} path with delay $d_{l_m}(\alpha)$, $m = 1, 2, ..., n$, in a randomly chosen path set $\mathcal{M}_n(\alpha) \in \mathcal{M}$. Using combinatorial theory, in the range $m \leq l \leq N - n + m$ this probability is given by

$$p_l(m) = \frac{1}{a_{\mathcal{M}}} C_{l-1,m-1} C_{N-l,n-m}. \tag{2.22}$$

The detailed derivation of Eq.(2.22) is in Appendix. Using Eqs. (2.21)-(2.22), the expected value of path delay of the m^{th} path of a randomly chosen path set $\mathcal{M}_n(\alpha) \in \mathcal{M}$ yields,

$$\begin{aligned} E\{d_m(\alpha)\} &= \sum_{l=m}^{N-n+m} d_l \cdot p_l(m) \\ &= \frac{1}{a_{\mathcal{M}}} \sum_{l=0}^{N-n} d_{(l+m)} C_{l+m-1,m-1} C_{N-l-m,n-m}, \end{aligned} \tag{2.23}$$

where $m = 1, 2, ..., n$. Thus, the expected maximal and minimal path delays over all path combinations are given as $E\{d_{\max}(\alpha)\} =$

$E\{d_n(\alpha)\}$ and $E\{d_{\min}(\alpha)\} = E\{d_1(\alpha)\}$, respectively. Then, using Eq. (2.23), FCT for random circuit switching can be derived as

$$\bar{\mathcal{T}}' = \mathcal{T}'(\alpha) = \frac{ML_M}{nC} + E\{d_n(\alpha)\}. \tag{2.24}$$

Finally, with Eqs. (2.20) and (2.23), the expected value of skew is

$$\bar{\tau} = E\{\tau(\alpha)\} = \frac{1}{a_{\mathcal{M}}} \sum_{i=0}^{N-n} (d_{(N-i)} - d_{i+1}) C_{N-i-1,n-1}. \tag{2.25}$$

Generally, each element d_l from $\vec{d}$ can have value equal to values of their direct neighbors, i.e., d_{l-1} and d_{l+1}. Thus, vector $\vec{d}$ can contain $D_{\min}$ and $D_{\max}$ elements with minimum and maximum path delays, i.e., $d_1 = d_2 = ... = d_{D_{\min}}$ and $d_N = d_{N-1} = ... = d_{N-D_{\max}+1}$, respectively. Therefore, there may be many more path combinations $\mathcal{M}_n(\alpha)$ yielding a maximum skew defined as follows

$$\tau_{\text{up}} = d_N - d_1 \equiv d_{N-D_{\max}+1} - d_{D_{\min}}. \tag{2.26}$$

The occurrence probability of τ_{up} is determined as follows

$$P_{\text{up}} = \sum_{j=max\{\substack{N-D_{\max}+1,\\ n\}}}^{N} \sum_{i=1}^{min\{\substack{D_{\min},\\ j-n+1\}}} P_i \cdot P_j \cdot \sum_{\alpha=1}^{C_{j-i-1,n-2}} P'(\alpha, n-2), \tag{2.27}$$

where for a path combination $\mathcal{M}_n(\alpha)$ we assume, that the k^{th} path $\mathcal{P}_{l_k}$ with maximal delay is mapped to one of the $D_{\max}$ paths $\mathcal{P}_j \in G_{\mathcal{P}}$, i.e, $d_{l_k}(\alpha) = d_j$ for $j = N - D_{\max} + 1, ..., N$. But if $n > N - D_{\max} + 1$, the mapping of the path with maximal delay has to be limited to $j \geq max\{n, N - D_{\max} + 1\}$. Furthermore, its path with minimal delay, $d_{l_1}(\alpha)$, is mapped to one out of $D_{\min}$ paths $\mathcal{P}_i \in G_{\mathcal{P}}$ for $i = 1, 2, ..., D_{\min}$. Otherwise, for a selected longest path $\mathcal{P}_j$, the first path of a set of n path is restricted to paths $\mathcal{P}_i \in G_{\mathcal{P}}$ with index $i \leq j - n + 1$, thus the mapping of $d_{l_1}(\alpha) = d_i$ is restricted to the range $i \leq min\{D_{\min}, j - n + 1\}$. Finally, there are $C_{j-i-1,n-2}$ possible path combinations, whose probability

$P'(\alpha, n-2) = (P_{\mathcal{M}})^{n-2}$ follows from Eq. (2.12) and is independent of α. Furthermore, the shortest and longest paths in each combination $\mathcal{M}_n(\alpha)$, i.e., one or more paths with delays d_1 and d_N from $\vec{d}$, respectively, occur with the same probability $P_i = P_j = P_{\mathcal{M}}$ in all combinations. Thus, Eq. (2.27) can be simplified as follows

$$P_{\text{up}} = \frac{1}{a_{\mathcal{M}}} \sum_{j=max\{\substack{N-D_{\max}+1,\\ n\}}}^{N} \sum_{i=1}^{\substack{min\{D_{\min},\\ j-n+1\}}} C_{j-i-1,n-2}. \tag{2.28}$$

When delay vector $\vec{d}$ contains only two elements with maximum and minimum delay, i.e., $D_{\min} = 1$ and $D_{\max} = 1$, the summation is omitted in Eq. (2.28), which yields

$$P_{\text{up}} = \frac{C_{N-2,n-2}}{a_{\mathcal{M}}} = \frac{C_{N-2,n-2}}{C_{N,n}} = \frac{n(n-1)}{N(N-1)}. \tag{2.29}$$

Random Packet Switching

Since the random packet switching method randomly selects each outgoing link e_j in any intermediate node v, the occurrence probability of e_j is the same for any outgoing link in the node v and defined by Eq. (2.13). As a result, each packet is sent over a randomly changing sequence of available links in the network so that the packets of a certain flow are randomly spread over all Ψ existing paths towards a destination. Thus, all Ψ existing available paths are always utilized, whereby some forwarding nodes v have more input interfaces than available output interfaces. Thus, some packets cannot be forwarded immediately and must be buffered with probability $P_B(v)$ defined by Eq. (2.10). Since Eq. (2.11) already considers the fact that more than one packet can claim the same link at the same time in any forwarding node, the FCT $\mathcal{T}^{\text{p}}$ of certain flow is a function of the end-to-end delay d_Ψ of the longest path $\mathcal{P}_\Psi$ utilized for

transmission. As a result, the ideal FCT $\mathcal{T}^{\mathrm{p}}$ is defined as:

$$\mathcal{T}^{\mathrm{p}} = \frac{ML_M}{nC} + d_\Psi. \tag{2.30}$$

Generally, the probability P_{loss}, that a sent packet is dropped by a forwarding node due to buffer overflow, can be calculated based on the standard queuing theory [46]. The packet dropping, however, results in a requirement for its retransmission and an additional increase of FCT. When the source data is split into M packets, the probability P_r that at least one sent packet is dropped and retransmitted is defined as $P_r = 1-(1-P_{loss})^M$. Then, the resulting FCT, which take into account the packet retransmission, is

$$\mathcal{T}^{\mathrm{p}}_{\sum} = (1-P_r)\mathcal{T}^{\mathrm{p}} + \gamma P_r \mathcal{T}^{\mathrm{p}}, \tag{2.31}$$

where $\gamma > 1$ is a proportion of FCT to take into account the delay due to retransmission. In contrast to the circuit switched networks, each specific path $\mathcal{P}_l$ between s and d has a certain occurrence probability P_l, which differs from the occurrence probability of other paths. The occurrence probability P_l of the path $\mathcal{P}_l$ can be defined by Eq. (2.8) and decreases with an increasing number of traversed intermediate nodes. Taking into account the occurrence probability of n utilized paths, the expected skew for a parallel transmission over $n = 2$ random paths can be calculated as follows

$$\bar{\tau}^p = \sum_{i=1}^{\Psi}\sum_{j=n}^{\Psi}(d_j - d_i)P_i \cdot P_j. \tag{2.32}$$

For transmission over $n > 2$ parallel paths, the skew is given as

$$\bar{\tau}^p = \sum_{i=1}^{\Psi}\sum_{j=n}^{\Psi}\sum_{\alpha=1}^{C_{j-i-1,n-2}}(d_j - d_i)P_i \cdot P_j \cdot P'(\alpha, n-2), \tag{2.33}$$

where the end-to-end delays and the corresponding paths occurrence probabilities are considered, while d_j and P_j as well as d_i and P_i

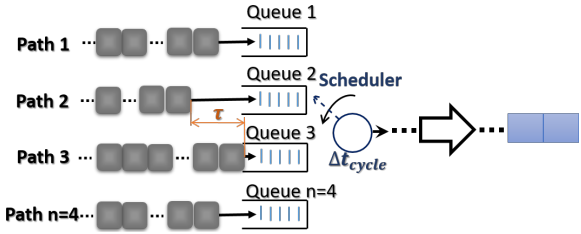


Figure 2.3: Deskew buffer models [1].

describe the randomly selected longest and shortest paths, respectively. Here, $P'(\alpha, n-2)$ describes occurrence probability of paths combination $\mathcal{M}'_{n-2}(\alpha)$, where $n-2$ paths have the end-to-end delays larger than or equal to d_i and less than or equal to d_j, i.e $\forall \mathcal{P}_l : \mathcal{P}_l \in \mathcal{M}_{n-2}(\alpha) : d_i \leq d_l \leq d_j$. The upper bound of skew is defined by Eq. (2.26) and occurs with probability P_{up} described by Eq. (2.27), where the number of available paths in circuit-switched networks, i.e., variable N, is replaced by the number of available paths in the packet-switched network, i.e., by variable Ψ.

2.3.4 Modeling and Analysis of De-Skew Buffer

Next, a closed form of expected and maximum de-skew buffer size, for an arbitrary distribution of path delays is analyzed.

The deskew buffer is presented in Fig. 2.3, which was already discussed at the beginning of Sec. 2.3. We assume that, in steady-state, the traffic is at maximum, with a deterministically distributed arrival process defined as one data block arrival per tu per lane. This assumption can be justified by the fact that n paths are selected to transport a single Ethernet frame and data blocks will arrive in succession on each of the parallel lanes in the receiver. We do not consider the impact of the polling strategy of the scheduler on buffer size. The assumption of an idealized scheduler is rather common,

and, for n input lanes, it means that the scheduler runs n time faster, such that during a full cycle time Δt_{cycle} a total of n data blocks are forwarded and processed every tu, i.e.,

$$\Delta t_{cycle}(n) = n \cdot (t_{\text{poll}} + t_{\text{fw}}) \leq 1[tu], \tag{2.34}$$

where t_{fw} is the mean forwarding time of a data block and $t_{\text{poll}} \ll t_{\text{fw}}$ is the polling and processing time.

Optimized Parallel Transmission

Here, the optimally chosen paths from set $\mathcal{M}_n(\alpha_{\text{opt}})$ result in minimal skew $\tau(\alpha_{\text{opt}}) = \tau$. In terms of buffer sizing, the receiver needs to deskew the skew τ given by Eq. (2.19) between the lanes with the largest and smallest delay. This delay, if measured in multiples of *time units*, requires a buffer size of (τ) for the shortest lane. The reordering and re-serialization process are undertaken by the scheduler, which requires an additional buffer place $1[tu]$ per lane to enable the processing after the cycle delay $\Delta t_{cycle}(k)$. The latter is a simple model for Ethernet's multi-lane deskewing mechanism using data block markers. Thus, the queue size required for the lane related to the shortest path is $(\tau + 1)$. To allow for any arbitrary pattern of skew, the input buffer size is the same for each lane, which corresponds to a classical design principle used in parallel hardware architectures. Thus, the total buffer size can be determined as

$$\Omega_{\text{ML}} = n \cdot (\tau + 1)\,. \tag{2.35}$$

Random Circuit Switching

The buffer analysis of the routing based on random path selection can be considered as an evaluation of buffer size for any possible skews. When no optimization is applied to routing and data blocks are sent over any $n = k$ out of N disjoint paths, which results in variations of skew. The expected value of the queue size can be

derived in closed form with Eq. (2.25) and Eq. (2.35). Thus, the average size of the deskew buffer in circuit switched networks is

$$\Omega_{\mathrm{ML}}^{\mathrm{R}} = E\{k \cdot (\tau(\alpha) + 1)\} = k \cdot (\bar{\tau} + 1)\,. \tag{2.36}$$

Random Packet Switching

The required deskew buffer in packet switched networks, $\Omega_{\mathrm{ML}}^{\mathrm{p}}$, can be derived by considering the mean skews $\bar{\tau}^{\mathrm{p}}$ defined by Eqs. (2.32) and (2.33). Since packets can arrive at the destination d on any $|\mathcal{E}_{in}(d)|$ incoming links resulting in the maximum arrival rate $\omega = |\mathcal{E}_{in}(d)|$ packets per *time unit*, there is a need for ω lane buffers. As a result, the total size of deskew buffer can be determined as follows

$$\Omega_{ML}^{\mathrm{p}} = \omega \cdot (\bar{\tau}^{\mathrm{p}} + 1)\,. \tag{2.37}$$

Analysis of Upper and Lower Bounds of Deskew Buffer

Next, we derive the absolute lower and upper bounds of a deskew buffer for arbitrary path sets and topologies. Since the deskew buffer for each lane has the same size, the lower and upper bounds of the deskew buffer will be the same for the maximum skew τ_{up} given in Eq. (2.26). Thus, for any arrival patterns of n data flows, the upper bound of the deskew buffer can be calculated as follows

$$\Omega_{\mathrm{up}} = \Omega_{\mathrm{low}} = n(\tau_{\mathrm{up}} + 1). \tag{2.38}$$

The maximum number of data blocks $\Omega_{\mathrm{up}}^{\mathrm{R}}$ needs to be buffered at receiver in circuit switched network, however, can be derived by considering all N available paths with the corresponding path delays. The worst case occurs when $n-1$ data flows with delays $d_1 \leq d_2 \leq \ldots \leq d_{n-1}$ from $\vec{d}$ arrive and need to be buffered at the receiver, while the n^{th} data block required for reordering start always takes a path with the largest path delay d_N. This path set is denoted as $\mathcal{M}_n(\alpha_{\mathrm{up}})$ and with $\tau_m(\alpha_{\mathrm{up}}) = d_{N_n}(\alpha_{\mathrm{up}}) - d_{l_m}(\alpha_{\mathrm{up}})$, where $d_{l_m}(\alpha_{\mathrm{up}})$

can be $d_{1_1}(\alpha_{\text{up}}) \leq d_{2_2}(\alpha_{\text{up}}) \leq ... \leq d_{n-1)_{n-1}}(\alpha_{\text{up}}) \leq d_{N_n}(\alpha_{\text{up}})$ the upper bound is given as follows

$$\Omega_{\text{up}}^{\text{R}} = n + \sum_{m=1}^{n} \tau_m(\alpha_{\text{up}}). \tag{2.39}$$

The upper bound of the queue size at receiver in packet-switched networks, $\Omega_{\text{up}}^{\text{p}}$, can be derived by considering all Ψ available paths. Thus, the worst case can be defined as a case, where $\omega = |\mathcal{E}_{in}(d)|$ packets from different packet sets arrive every *time unit* at the destination d over ω shortest paths with delays $d_1 \leq d_2 \leq ... \leq d_\omega$ from $\vec{d}$ and must be buffered, while the n^{th} data block from each packets set required for start of frame recovery always takes the longest path $\mathcal{P}_\Psi$ with the largest end-to-end delay d_Ψ. This path set is denoted as $\mathcal{M}_\omega(\alpha_{\text{up}}^{\text{p}})$ and with $\tau_m(\alpha_{\text{up}}^{\text{p}}) = d_{\omega_n}(\alpha_{\text{up}}^{\text{p}}) - d_{l_m}(\alpha_{\text{up}}^{\text{p}})$, where $d_{l_m}(\alpha_{\text{up}}^{\text{p}})$ can be $d_{1_1}(\alpha_{\text{up}}^{\text{p}}) \leq d_{2_2}(\alpha_{\text{up}}^{\text{p}}) \leq ... \leq d_{\omega)_o}(\alpha_{\text{up}}^{\text{p}}) \leq d_{\Psi_n}(\alpha_{\text{up}}^{\text{p}})$, where $o \in [1, ..., n-1]$ from any packet set. As a result, the upper bound $\Omega_{\text{up}}^{\text{p}}$ is a queue size formed during two initial periods before the steady-state. The first period results in the queue size caused by out of order packet arrivals and before the packets on all ω shortest paths $\mathcal{P}_l$ with delays $d_l \in \vec{d}$, $l \in [1, \omega]$, reached the destination. The second period causes an additional queue built in the period between two events, i.e., all ω input interfaces at receiver deliver for the first time ω packets per *tu* and the first frame recovery.

$$\Omega_{\text{up}}^{\text{p}} = \sum_{m=1}^{\omega} \tau_m(\alpha_{\text{up}}^{\text{p}}) + \omega \cdot \lceil d_\Psi - d_\omega \rceil. \tag{2.40}$$

2.4 Modeling and Analysis of Parallel Transmission with Coding

To address some challenges of parallel transmission, an erasure code such as random Linear Network Coding (LNC) can be applied to the parallelized traffic. Erasure codes can add redundancy into data to protect it against losses by erasure correction. Our choice of LNC is motivated by the potential use of the recoding capability and design of a unified code for a network system in a cross-layer fashion. With recoding, it is possible to insert additional redundancy without the need for decoding, and so further increase fault tolerance. Even after repeated coding (recoding), the effect upon the data is of a single linear transformation, requiring no decoding in the network, but a single linear inversion at the destination. That presents the key features in the multi-layer design. However, we assume, in this section, that encoding and decoding are applied in the end-systems of the circuit-switched network, i.e., on the Ethernet layer, while intermediate network nodes just forward the incoming coded data over pre-reserved network interfaces. This section investigates the impact of coding on a skew and de-skew buffer.

2.4.1 Erasure Coding Approach

A LNC can be applied on the parallelized data sequences as illustrated in Fig. 2.4, where any traffic sequence is decomposed into *data blocks* of the same length, i.e., $\mathfrak{h}$ symbols each. Since a coding process is performed over a finite field F_{2^b}, each symbol has the same length of b bits. Any k blocks from k parallel lanes build a so-called *generation* and are encoded with the same set of coding coefficients into n blocks, whereby k defines a *generation size* and n defines a *coded generation size*. The number of resulting encoded blocks n after encoding (also corresponding to the number of network paths) can be equal or larger than the generation size, i.e., $n \geq k$. If $n > k$,

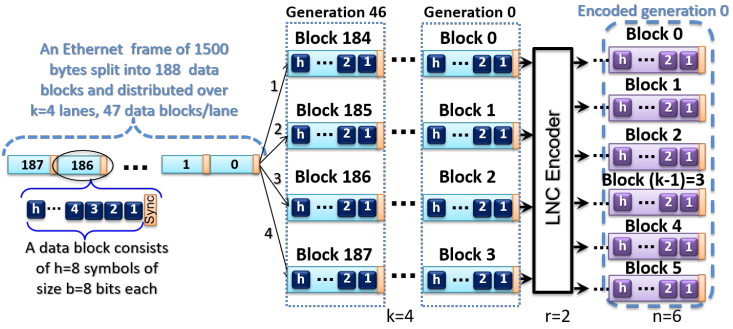


Figure 2.4: Ethernet traffic parallelization and random LNC [1].

we refer to $r = n - k$ blocks as *redundancy*.

In Fig. 2.4, the Ethernet frame provides 47 generations, whereby $k = 4$ data blocks are encoded so that each generation is extended by $r = 2$ redundant blocks resulting in $n = 6$ blocks per coded generation. We refer to the number of resulting parallel lanes required after encoding, i.e., n, as the *level of parallelism* in an end-system.

The decoding is only possible, when at least k encoded data blocks from the same coded generation arrived and can be decoded, e.g., by running Gaussian elimination. In the case of data block loss or network failures, the redundant data blocks can replace any data block from the same generation. Later, we show that this feature is useful for not only fault tolerance but also for reducing the expected value of the skew and, thus, the buffer size at the receiver. It should be noted, that random LNC decodability can be challenging if there are linearly dependent combinations in the code, whereby the probability of such combinations is related to the finite field size selected. For instance, the probability of selecting coefficients that do not correspond to a decodable code instance is of the order of 2^{-8} [47, 48, 49, 50, 51, 52].

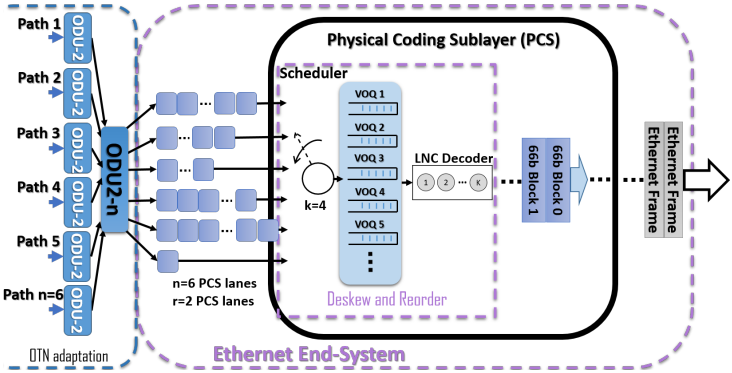


Figure 2.5: Receiver decoding buffer [1].

2.4.2 Design of Parallel Transmission with Coding

A nice feature of random LNC is that it can be implemented without altering the system architecture previously presented in Fig. 2.1(a), whereby LNC encoder can be integrated into Ethernet Layer directly after traffic parallelization. In Fig. 2.4, the exemplary Ethernet frame of 12000 bits (1500*bytes*) is split into 188 data blocks and then encoded with 64b/66b line code. The line code is not to be mistaken for a random LNC. The latter is performed over the last 64b after sync header. We introduce the notation 64b/64b+2b to differentiate between 64 data bits and the 2 bits of the sync header according to [2]. The 2 sync header bits of each data block bypass the random LNC encoder and are added after encoding as a header in the form 64b/64b+2b+Cb, where C is an additional ID-header as we will discuss later. When the coding process is based on symbol size $b = 8$ bits and data blocks contain 64 data bits excluding the 2 synchronization bits, each data block would contain $\mathfrak{h} = 8$ symbols. The data blocks are then distributed over $k = 4$ virtual lanes so that each sub-flow on a defined virtual lane contains exactly 47 data blocks. In our model of traffic splitting, we assume the bit padding

in that case that the data block, or symbol, is incomplete, e.g. 32 bit in our example. The LNC encoder takes one data block from each lane, in total k data blocks, and encodes them together into n coded data blocks. All n blocks belong to the same coded generation and send in parallel over n paths toward the destination.

Now, let us consider the reference deskew and reorder model in the receiver with random LNC shown in Fig. 2.5, where transmission is implemented over $n = 6$ paths. The distributed line buffer of the Ethernet system is now organized as a centralized decoding buffer consisting of multiple virtual output queues (VOQ). A new VOQ is created for each generation, each time the first data block of a new generation arrives. The decoder checks the existing VOQ for complete generations and starts decoding as soon as one generation is complete. All data blocks of a complete generation are decoded in parallel, by running Gaussian elimination. That replaces the line-specific deskewing approach of the Ethernet system by taking advantage of the multiplexing gain due to the centralized buffer. After decoding, the data blocks are sent in the correct order, – thus eliminating the need for reordering. That is because a correct assignment of decoding coefficients to encoded data blocks assures the right order of data blocks.

For successful decoding on the Ethernet layer, all data blocks from the same generation g_ν need to be uniquely identified, which can be implemented using an additional $C = 6$ bits in the header to form 72b coded blocks. Then, the 72b=64b/64b+2b+6b block increases the line rate by 12/11, which requires to develop an efficient Generic Framing Procedure method [53]. Each data block of a generation g_ν is identified by the same number $b_{\nu,j} = l, j = 1, ..., n$, where $0 \leq l \leq L$ and $L = 2^C$. At the receiver, the identifier is processed by the scheduler and addresses the correct VOQ l. Since ODU2 payload includes 15232 bytes, after $L = 64$ sequentially arrived data blocks per lane, the number wrap-round will overwrite the same VOQ with

a new generation. That corresponds to the maximum delay difference between lanes to $64tu$. Thus, the parallelization, reordering and de-skewing at the end-systems can be modeled as a discrete-time process. For instance, for 10Gbit/s and $tu = 7.2$ns, a maximum delay difference is 446.4 ns, which is far larger than the required 180ns for Ethernet systems, and thus would be unacceptable.

In circuit-switched networks, it is possible to reuse the Ethernet inherent alignment marker process without adding additional ID-header bits to data blocks, i.e. $C = 0$, also in line with the existing standards. After receiving the first alignment marker, the corresponding lane is marked as a reference, the scheduler initialized the VOQ 1 for the first generation and each following block of the same line the next VOQ l is generated. In fact, this allows us to address $m_L = 16383$ different VOQs and to compensate delay differences of up to $16383tu$. To limit the buffer space, however, similar to the method with ID-header, the VOQs can be cyclically overwritten after receiving the L^{th} data blocks on the initializing lane, where $L << m_L$. In general, the data blocks extracted from an ODU container or its sync header may be erroneous, despite the existence of FEC in the optical layer. In case of a single block error, the random LNC decoding of one generation will fail, if no coding redundancy is introduced, i.e., $n = k$. If an alignment marker is erroneous, all data blocks received on the associated lane may be sorted to the wrong VOQ resulting in decoding errors. This error process will be stopped with the arrival of the next marker and, thus, re-initialization of VOQs mapping. At the same time, this issue is not different from the erroneous packet handling in the conventional Ethernet system.

The coding coefficients required for encoding and decoding can be selected and distributed by the control plane using an out-of-band signaling channel. An in-band signaling method can be used applying transcoding, as specified in IEEE802.bj. For example with 4 PCS lanes, the 256b/257b transcoding enables us to transmit 35

additional bits after sending 5 blocks of 256b/257b (or after 20 blocks of 64b/66b) per lane (in total, 140 additional bits serially) without increasing the 10GE line rate, which is in line with the Ethernet and OTN standards. Thus, $k = 4$ coding vectors each of length 8 bit can be sent on each lane every 20 64b/66b blocks to the receiver, i.e., 20 successive generations are coded with the same coding vector. The overhead due to exchange of coding coefficients can be significantly reduced by a pseudo-random number generator (pRNG) utilized in sender and receiver. The use of pRNG allows the generation of an infinite number of pseudo-random coding coefficients in sender and receiver, whereby only a short start sequence needs to be exchanged.

From the network perspective, the end-system splits a serial flow of high-speed data blocks into k parallel flows (sub-flows, or lanes), which are to be coded into n flows using a random LNC encoder. After coding, each parallel flow is independently routed over n separate optical paths, whereby $n \geq k$. We assume that the aggregated capacity in the network is always greater or equal to the capacity of the end-systems. Thus, also for the case $n > k$, the number of encoded data flows n can match the number of parallel network paths and interfaces requiring $r = n - k$ redundant paths.

2.4.3 Analysis of Coding Overhead

For the Ethernet-over-optical network, the optical transmission impairments such as chromatic dispersion and polarization mode dispersion, which typically requires a higher optical signal to noise ratio (OSNR), need to be considered [54]. These transmission impairments cause bit errors, which complicate the correct detection of signals at the receiver. Faults in the optical layer can lead to packet errors on the Ethernet layer and the network layers, thus, degrading the overall system performance, i.e., leading to an increase in FCT and a skew. Fault tolerance has been subject to significant past research, most notably in the area of Forward Error Correction

(FEC) codes, which are commonly applied in the physical layer to ensure an end-to-end bit error rate. However, there is comparably less insight into the situation, when bit errors arose in the physical layer cannot be covered by the mechanisms, such as FEC; and their impact on the packet errors in the higher layers. That is especially critical in dynamically routed networks, where the proposed physical layer schemes under varying network conditions and optical path lengths cannot always guarantee the correct reception of packets. The design of end-systems in terms of robustness and tolerance to transmission impairments is becoming even more critical. Since the proposed coded parallel transmission system applies to the Ethernet layer above the physical layer, it complements (but does not replace) FEC-based physical layer mechanisms to mitigate bit errors.

Without loss of generality, we conservatively define that if any bit in a packet can not be correctly detected (even if for one faulty bit only), the packet is considered as erroneous and dropped. That is usually referred to as *packet loss*. Hence, Bit Error Rate (BER) can be directly translated into the packet error rate (PER) and indirectly interpreted as a measure of that packet loss precisely. The relation between PER and BER can then be defined as follows:

$$PER = 1 - (1 - BER)^{L_M}, \tag{2.41}$$

where L_M is the size of an Ethernet data block in bits. To measure the system performance in terms of fault tolerance, we define a parameter *Acceptable Packet Loss (APL)*, as the number of packets that can be lost, or contain errors after the transmission, without affecting the system performance in terms of correct packet reception. When the number of erroneous or lost packets (data blocks) is smaller than APL, the system can correctly receive and recover the original Ethernet frame and is considered as *fault tolerant*.

We now model the bit error rate p'_e, which occurs in the physical layer during transmission. Generally, the expected BER on the parallel paths is a function of a modulation level and OSNR [55, 56, 57].

To detect and to correct bit errors occurred in the physical layer, an FEC code with a code rate R is applied to L_M information bits, which is the size of an LNC encoded data block in the Ethernet layer. We assume that each data block is encoded with FEC code independently of other data blocks from the same Ethernet frame, which allows independent and timely processing of data blocks at the receiver without increasing a queuing delay. As a result, the size of each data block after FEC encoding is $\frac{L_M}{R}$ bits, which are transmitted over the optical network and affected by bit errors.

We do not specify any FEC code and just assume the worst case for the sake of our analysis, i.e., the simplest possible FEC code with the lowest complexity encoding and decoding, where the minimal Hamming distance is $h_{min} = \frac{L_M}{R} - L_M$. Generally, the FEC code can detect and correct multiple bit errors. The number of erroneous bits that can be detected and corrected by a FEC code is denoted as t_e, $t_e = h_{min} - 1$, and t_k, $t_k = \frac{h_{min}-2}{2}$, if h_{min} is an even number, respectively. Moreover, all erroneous LNC coded data blocks of size L_M are marked after FEC and will be replaced during the LNC decoding process. Then, the expected residual error probability p_r can be predicted as $p_r = \sum_{j=t_e+1}^{\frac{L_M}{R}} \binom{\frac{L_M}{R}}{j} p_e'^j (1-p_e')^{\frac{L_M}{R}-j}$, where p_e' is a bit error rate on used parallel paths in optical network. Thus, the recovery of an Ethernet frame with LNC is successful with probability $(1-p_r)$, when there are at most t_e erroneous bits in the frame, i.e., $\frac{L_M}{R} \cdot p_e' \leq t_e$. As the proposed LNC coded parallel transmission is a method complementary to the FEC already in place in the physical layer, the expected BER before LNC decoding is reduced as

$$BER = \frac{L_M \cdot p_e' - R \cdot t_k}{L_M}, \tag{2.42}$$

where t_k out of $\frac{L_M}{R} \cdot p_e'$ initial bit errors are corrected by FEC and L_M is a total number of bits in each data block sent to LNC decoder.

To ensure a fault-tolerant transmission, k data blocks encoded into n LNC coded data blocks are sent over the parallel path, whereby at

least $APL \cdot n = r$ redundant data blocks are required. As a result, the APL can be expressed as follows

$$APL \geq \frac{r}{n}. \tag{2.43}$$

However, to guarantee the correct reception of all data blocks at the receiver, the acceptable packet loss must be at least equal to the packet error rate, i.e., $APL \geq PER$.

Let us assume that the proposed parallel transmission system is able to adapt LNC parameters, i.e., k, r and n, to channel conditions, i.e., to bit error rate BER as well as a code rate R of FEC code to p'_e. Next, we analyze the transmission overhead due to transmission of redundant LNC coded data blocks in addition to FEC redundancy, where packet size is L_M before FEC. Here, we assume that all parallel paths are affected by the same bit error rate p'_e and all LNC coded packets are protected with FEC in the same manner and the same code rate R_{LNC}. However, without LNC, the code rate of the FEC code is denoted as R.

Lemma 2.1 *The proposed parallel transmission system with LNC can tolerate bit error rate p'_e up to* 100% *and increases the code rate of an arbitrary FEC code, i.e., $R_{LNC} > R$.*

The proof of Lemma 2.1 is presented in Appendix 7.2.

Without loss of generality, sending of r redundant data flows over additional r parallel paths results in a transmission overhead, which we define and numerically evaluate as follows

$$\boldsymbol{\Theta_t} = \frac{r}{k}. \tag{2.44}$$

To evaluate an overhead due to additional redundant bits generated by LNC and FEC, let us define the overall information flow Φ as a total number of original information bits, i.e., $L_M \cdot k$, divided by the

total number of bits provided after encoding to be sent in parallel to the destination, i.e., $\frac{nL_M}{R}$. Then, the overall information flow is

$$\Phi = \frac{L_M kR}{nL_M} = \frac{kR}{n}, \tag{2.45}$$

where $n := k$ and $R := R$ in the case of transmission without LNC and $n \geq k$ $R := R_{LNC}$ in parallel transmission system with LNC. Than, the total transmission overhead Θ can be defined as follows

$$\Theta = 1 - \Phi = \frac{n - kR}{n}. \tag{2.46}$$

Lemma 2.2 *The total transmission overhead Θ_{LNC} of the parallel transmission system with LNC and FEC is less than the transmission overhead Θ of the parallel transmission system with FEC only if the expected bit error rate on the parallel transmission paths is determined as $p'_e > \frac{1}{3}$.*

The proof of Lemma 2.2 is presented in Appendix 7.3.

2.4.4 Combinatorial Delay Analysis

This subsection provides a derivation of the expected value of the skew in arbitrary circuit-switched networks with coding and random routing, the analysis of occurrence probability of maximum skew, analysis of the impact of path failures and packet loss on the skew.

As previously mentioned, k parallel lanes can be coded into $n = k + r$ data blocks and thereafter transmitted over n paths. The receiver needs to buffer only any k out of $n = k + r$ data blocks from the same generation for the decoding start, while any r data blocks arriving later can be ignored. Thus, in a parallel network system with random LNC, the sender can in fact arbitrarily assign any of the n paths, irrespectively of their related path delays, which eliminates the need for path optimization. The analysis of that parallel system is not straightforward and needs a few modified expressions of the expected values of skew.

Let us assume that all $n = k + r$ parallel paths are selected randomly with the same probability. For each path set $\mathcal{M}_n(\alpha)$ the corresponding delays are sorted as $d_{l_1}(\alpha) \leq d_{l_2}(\alpha) \leq ... \leq d_{l_k}(\alpha) \leq d_{l_{k+1}}(\alpha) \leq ... \leq d_{l_m}(\alpha) \leq ... \leq d_{l_{k+r}}(\alpha)$, $d_{l_m}(\alpha) = \sum_i d_i \delta_\alpha(i,m)$, d_i from $\vec{d}$, $1 \leq m \leq n$. For successful decoding, the receiver needs to buffer the data blocks arriving from any k paths with least delays, i.e., $d_{l_m}(\alpha), m = 1, 2, ..., k$.

In a loss-free parallel network system with r redundant paths, where the path set $\mathcal{M}_n(\alpha)$ of $n = k + r$ paths are randomly chosen, a maximal path delay which has to be taken into account at the decoder is bounded as follows

$$d_k \leq d_{\max}(\alpha, r) \leq d_{N-r}. \tag{2.47}$$

The detailed derivation of Eq. (2.47) is in Appendix in Sec. 7.1. With Eqs. (2.47), (2.22), (2.23), an expected maximal path delay is

$$E\{d_{\max}(\alpha)\}(r) = \sum_{l=k}^{N-r} p_l(k) \cdot d_l = E\{d_k(\alpha)\} = E\{d_{n-r}(\alpha)\}. \tag{2.48}$$

Similarly, the expected minimal path delay can be calculated as $E\{d_{\min}(\alpha)\}(r) = E\{d_1(\alpha)\}$. Finally, with Eq. (2.48) and using the fact that the number of paths is now given by $n = k+r$ the expected value of skew follows to be

$$\bar{\tau}(r) = E\{d_{n-r}(\alpha)\} - E\{d_1(\alpha)\}, \tag{2.49}$$

where $\bar{\tau}(0) = \bar{\tau}$ leads to the result given by Eq. (2.25).

As a result, the maximum skew is determined as follows

$$\tau_{\text{up}}(r) = d_{N-r} - d_1. \tag{2.50}$$

Due to the fact, that the delay vector $\vec{d}$ can contain $D_{\max}$ and $D_{\min}$ elements d_l, which are equal to d_N and d_1, respectively, it is advantageous to consider a special case covered by Eq.(2.50). If $r < D_{\max}$ the maximal delay considered is given by $d_{N-D_{\max}+1} = d_{N-r} = d_N$,

and $\tau_{\mathrm{up}}(r) = d_{N-D_{\max}+1} - d_1$ is independent of r. Thus, an equivalent form of Eq.(2.50) is given by $\tau_{\mathrm{up}}(r) = d_{N-max\{r,D_{\max}-1\}} - d_1$. Using this result and similar to Eq. (2.27), the occurrence probability of maximum skew $\tau_{\mathrm{up}}(r)$ is given by Eq. (2.51).

$$P_{\mathrm{up}}(r) = \sum_{\substack{j=max\{N-max(r,D_{\max}-1),\\ n-r\}}}^{N-r} P_j \sum_{i=1}^{\substack{min\{D_{\min},\\ j-n+r+1\}}} P_i \cdot \cdot \sum_{\beta=1}^{C_{N-j,r}} P'(\beta,r) \sum_{\alpha'=1}^{C_{j-i-1,n-r-2}} P'(\alpha', n-r-2). \tag{2.51}$$

With r redundant paths, only $k = n-r$ paths are effectively used and the maximum possible path delay is determined by $d_{N-max\{r,D_{\max}-1\}}$, as discussed above. Thus, in contrast to Eq. (2.27), the k^{th} path $\mathcal{P}_{l_k}$ with maximal delay is mapped to a path $\mathcal{P}_j \in G_{\mathcal{P}}$ within the range given by $d_{l_k}(\alpha) = d_j$ for $j = max\{n-r, N-max\{r, D_{\max}-1\}\}, ..., N-r$. Similar, the path with minimal delay, $d_{l_1}(\alpha)$, can be mapped to one of the paths $\mathcal{P}_i \in G_{\mathcal{P}}$ out of the range $i = 1, 2, ..., min\{D_{\min}, j-n+r+1\}$. In accordance with our assumptions, the r redundant paths have a delay larger or equal than the k^{th} path. Now, depending on the mapping of the k^{th} path to a path $\mathcal{P}_j$, the redundant paths $\mathcal{P}_{l_{k+1}}(\alpha), ..., \mathcal{P}_{l_{k+r}}(\alpha)$ will be mapped to paths $\mathcal{P}_l \in G_{\mathcal{P}}$ with delays d_l, where $l = j+1, ..., N$. To achieve this mapping, there are $C_{N-j,r}$ path combinations, which are combined in subset $\mathcal{M}_n(\beta)$. Due to Eq. (2.12), the probability of the set of r redundant path can be written as $P'(\beta, r) = (P_{\mathcal{M}})^r$. Similar to Eq. (2.27), for the remaining $n-r-2$ paths there are $C_{j-i-1,n-r-2}$ combinations whose paths are combined in subset $\mathcal{M}_n(\alpha')$, where the probability of a set of $n-r-2 = k-2$ paths can be written as $P'(\alpha', n-r-2) = (P_{\mathcal{M}})^{k-2}$. Thus, the probability of maximum skew, $P_{\mathrm{up}}(r)$, can be derived as follows

$$P_{\mathrm{up}}(r) = 1/a_{\mathcal{M}} \sum_{j=max\{N-\substack{max(r,D_{\max}-1),\\ n-r\}}}^{N-r} C_{N-j,r} \cdot \sum_{i=1}^{\substack{min\{D_{\min},\\ j-n+r+1\}}} C_{j-i-1,n-r-2}. \tag{2.52}$$

When there is only one path with maximum delay, d_N and only one path with minimum delay, d_1, i.e $D_{\max} = D_{\min} = 1$, the occurrence probability of a maximum skew is derived from Eq. (2.52) as

$$P_{\mathrm{up}}(r) = \frac{C_{N-r-2,n-r-2}}{C_{N,n}} = \prod_{i=0}^{r+1} \frac{(n-i)}{(N-i)}. \tag{2.53}$$

Compared to the system without coding redundancy, the receiver experiences *a reduction of a maximum skew* $\mathbf{\Delta}^{\max}$, expressed as

$$\mathbf{\Delta}^{\max} = \frac{d_N - d_{N-r}}{d_N}. \tag{2.54}$$

Additionally, the expected value of skew can be reduced by the ratio $\mathbf{\Delta}^{\tau}$ determined as follows

$$\mathbf{\Delta}^{\tau} = \frac{\bar{\tau} - \bar{\tau}(r)}{\bar{\tau}}. \tag{2.55}$$

Impact of Traffic Loss on resulting Skew

Coded parallel transmission does not only provide a reduction of skew but also can increase the robustness of the system regarding packet loss and path failures. That is possible when the number of lost data blocks f_{pack} per coded generation or number of failed paths f_{path} is less or equal to redundancy r. For random path selection, we now consider a possible impact of faults on the resulting skew.

Let us consider the worst case, where packet loss can occur on any utilized path, where $f_{\mathrm{pack}} \leq r$. Here, the maximal skew needs to be considered to prevent frame dropping. Thus, the mean skew can be calculated with Eq. (2.23) as follows

$$\bar{\tau}(r, f_{\mathrm{pack}}) = E\{d_n(\alpha)\} - E\{d_1(\alpha)\}. \tag{2.56}$$

Let us further consider a network model, where available paths can fail with equal probability, $f_{\text{path}} \equiv f$. When the path failures occur before a set of paths is selected for transmission, the source does not consider failed paths for transmission and selects n paths out of the remaining $N-f$ available paths as long as $N-f \geq n$. The expected skew can be calculated by Eq. (2.49) under consideration of a reduced delay vector $\vec{d}$, which contains now $N-f$ elements.

When the path failures, however, occur after a set of n paths is already selected and utilized, the data transmission is only successful if at most r paths fail, i.e., $f \leq r$. In particular, random path selection allows to analyze the effect of the failure pattern in two possible ways: 1) any f paths out of N available paths from a set $G_{\mathcal{P}}$ can fail during data transmission and, as a result, e paths can fail over any randomly selected path combination $\mathcal{M}_n(\alpha)$; 2) f paths exactly fail in any randomly selected set of paths $\mathcal{M}_n(\alpha)$.

In the first scenario, we derive an average value of skew over all possible path sets. Let Ξ, $|\Xi| = b_\Xi = C_{N,f}$, be a set of all possible combinations of f failed paths out of N available. Then, each combination presents a subset $\Xi_f(\beta) = \{\mathcal{P}_{l_1}(\beta), ..., \mathcal{P}_{l_\xi}(\beta), ..., \mathcal{P}_{l_f}(\beta)\}$, $\xi = 1, 2, ..., f$, $\Xi_f(\beta) \in \Xi$, $1 \leq \beta \leq b_\Xi$, which is a sorted subset of set $G_{\mathcal{P}}$ and appears with same probability $P(\Xi_f(\beta)) = 1/b_\Xi$. Also here, any path $\mathcal{P}_{l_\xi}(\beta)$ corresponds to a path $\mathcal{P}_l \in G_{\mathcal{P}}$. Then, for each combination β of failed paths, the set $G_{\mathcal{P}}$ is reduced to $G^\beta_{\mathcal{P}} = G_{\mathcal{P}} \backslash \Xi_f(\beta)$ with related sorted delays $\vec{d}^{\,\beta} = (d^\beta_1, d^\beta_2, ..., d^\beta_{N-f})$. Similar, each set $\mathcal{M}_n(\alpha)$ is reduced to a sorted set $\mathcal{M}^\beta_n(\alpha)$, $\mathcal{M}^\beta_n(\alpha) = \mathcal{M}_n(\alpha) \backslash \Xi_f(\beta) = \{\mathcal{P}^\beta_{l_1}(\alpha), ..., \mathcal{P}^\beta_{l_m}(\alpha), ...\}$, where $n - f \leq |\mathcal{M}^\beta_n(\alpha)| \leq n$. The delay $d^\beta_m(\alpha)$ related to path $\mathcal{P}^\beta_{l_m}(\alpha)$ is mapped to the delay in $\vec{d}^{\,\beta}$. The impact of path failures on resulting delays can be analyzed using Eq. (2.23) as discussed below. For f failed paths in $G_{\mathcal{P}}$ the number of available paths N is reduced to $N-f$. For a fixed failure pattern $\Xi_f(\beta)$, the amount of failed paths in any subset $\mathcal{M}_n(\alpha)$ is $e = (f-i), i = 0, ..., f$, yielding the set $\mathcal{M}^\beta_n(\alpha)$ with

$|\mathcal{M}_n^\beta(\alpha)| = n - e$ paths. This affects the number of combinations used in Eq. (2.23) by replacing n by $n - e = n - (f - i)$ and taking into account that the summation of delays d_{l+m}^β has to be adapted to the range $l = 0, ..., N - f - (n - (f - i)) = N - n - i$. Thus, the expected delay of the m^{th} path in any randomly selected path set $\mathcal{M}_n^\beta(\alpha)$ and over all possible failure pattern $\Xi_f(\beta)$ is

$$E_1\{d_m(\alpha, \beta, f)\} = \frac{1}{a_{\mathcal{M}} b_\Xi} \sum_{\beta=1}^{b_\Xi} \sum_{i=0}^{min\{f, N-n\}} C_{f,f-i} \sum_{l=0}^{N-n-i} d_{(l+m)}^\beta C_{l+m-1,m-1} C_{N-f-l-m,n-f+i-m}, \tag{2.57}$$

where binomial coefficient $C_{f,f-i} C_{N-f-l-m,n-f+i-m}$ takes into account the number of path sets $\mathcal{M}_n^\beta(\alpha)$ with $f - i$ failed paths, where m^{th} paths appears on position $l + m$ in $G_{\mathcal{P}}^\beta$. Then, with Eq. (2.49)-(2.57), the expected skew yields

$$\bar{\tau}_1(r, f) = E_1\{d_{n-r}(\alpha, \beta, f)\} - E_1\{d_1(\alpha, \beta, f)\}. \tag{2.58}$$

We now analyze the second scenario as a special case of the first one, whereby exactly f parallel paths fail in each path combination $\mathcal{M}_n(\alpha)$. Here, there are $C_{n,f}$ possible combinations of failed paths. Each set $\mathcal{M}_n(\alpha)$ can be then recombined into $C_{n,f}$ reduced path sets of only $n - f$ paths. On the other hand, there can be at most b_Ξ different combinations $\Xi_f(\beta)$ of failed paths over all N available paths and over all sets $\mathcal{M}_n(\alpha)$. Thus, there are $a_{\mathcal{M}}^f = C_{N,n-f}$ different path combinations $\mathcal{M}_{n-f}(\alpha) \equiv \mathcal{M}_n^\beta(\alpha)$, $|\mathcal{M}_n^\beta(\alpha)| = n - f$, which can deliver data blocks to the receiver in case f out of n randomly selected paths fail. The expected delay of the m^{th} path in any randomly selected path set $\mathcal{M}_n^\beta(\alpha)$ under failure pattern $\Xi_f(\beta)$ can be derived from Eq. (2.57) by considering $e = (f - i)$, $i = 0 =$

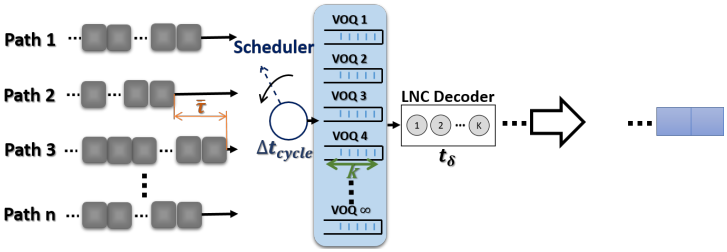


Figure 2.6: Decoding buffer model [1].

const and is given as follows

$$E_2\{d_m(\alpha,\beta,f)\} = \frac{1}{a_{\mathcal{M}} b_{\Xi}} \sum_{\beta=1}^{b_{\Xi}} \sum_{l=0}^{N-n} d^{\beta}_{(l+m)} \cdot \\ \cdot C_{l+m-1,m-1} C_{N-f-l-m,n-f-m}. \tag{2.59}$$

Thus, the expected skew is given with Eqs. (2.49) and (2.59) as

$$\bar{\tau}_2(r,f) = E_2\{d_{n-r}(\alpha,\beta,f)\} - E_2\{d_1(\alpha,\beta,f)\}. \tag{2.60}$$

2.4.5 Modeling and Analysis of Decoding Buffer

Next, a closed-form of expected and maximum size of decoding buffer (Fig. 2.6) is derived for an arbitrary distribution of path delays in circuit-switched networks and the impact of path failures and packet loss on the buffer size is analyzed.

The first data blocks from any n lanes and originated from the first generation typically arrive at different times at the receiver. To analyze the effect of skew for $n = k$ input lanes, we first assume a fixed path pattern $\mathcal{M}_n(\alpha)$ with delays $d_{l_m}(\alpha)$ ordered as $d_{l_1}(\alpha) \leq d_{l_2}(\alpha) \leq ... \leq d_{l_k}(\alpha)$. The scheduler has to poll k input lanes and as soon as the first data block of a new generation arrives and that on a lane corresponding to the shortest path, the scheduler forwards the data block to a newly created virtual output queue (VOQ). In the initial phase, the decoding commences when k data blocks of

the first generation arrive. Consequently, the data blocks from the same generation arriving from the $k-1$ shorter paths have to be buffered in the decoding buffer until the k^{th} data block from the same generation is received. Thus, before the last k^{th} packets from the first generation arrives, $\Omega_{\text{ini}}(\tau)$ packets from other $k-1$ lanes must be buffered. With Eqs. (2.16)-(2.17) , the queue size during the initial phase is thus determined as follows

$$\Omega_{\text{ini}}(\tau) = \sum_{m=1}^{k} \tau_m(\alpha) = \tau(k, \alpha), \tag{2.61}$$

where the explicit dependence on the path set index α is omitted. Note, as soon as there is no skew, i.e., $\tau = 0\ tu$, the initial size of the decoding queue defined by Eq.(2.61) becomes zero. That is because the data blocks are immediately transferred by the scheduler during the subsequent cycle.

For the deterministic arrival process, a steady-state is reached after the first generation completes. Here, the number of data blocks forwarded in every decoding interval t_δ can be calculated as follows

$$\Omega_\delta(t_\delta) = \left\lfloor \frac{t_\delta}{t_{\text{poll}} + t_{\text{fw}}} \right\rfloor. \tag{2.62}$$

In a steady-state, the decoding process finishes after the decoding interval t_δ, and all data blocks from one completed generation leave the decoding buffer. To avoid idle periods of decoder, a further decoding process must immediately commence after the previous decoding is finished. In the best case, the scheduler transfers a new generation to the decoder every Δt_{cycle}, i.e. $t_\delta \leq \Delta t_{cycle}$. On the other hand, one data block forwarded to a decoding buffer can complete one generation due to backlog in the case of $\tau > 0$. Consequently, one data block can be enough to trigger the next decoding process after the current decoding. For this purpose, the decoding interval t_δ can take a value in the range $\frac{\Delta t_{cycle}}{n} \leq t_\delta \leq \Delta t_{cycle}$. Although a short decoding interval t_δ will release full generations

very rapidly, new decoding can only start after a new generation has reached the decoding buffer, which in the worst case can last Δt_{cycle}. Thus, one can argue that the longest possible decoding interval is the best choice, and therefore we will assume $t_\delta = \Delta t_{cycle}$. As a result, the decoding queue size can be calculated by combining Eq.(2.61), the buffer needed due to skew τ, and Eq.(2.62), the number of data blocks transferred to the decoder in steady-state during the cycle time Eq.(2.34), given as, i.e.,[58]

We assume the worst case scenario, whereby decoding by Gaussian elimination has complexity $O(n^3)$, and analyze next the value of decoding interval t_δ. In steady state, the decoding process finishes after the decoding interval t_δ, and all data blocks from one complete generation leave the decoding buffer. To avoid idle periods, a new decoding process should immediately commence after previous decoding cycle is finished. In the best case, the scheduler transfers a new generation to the decoder every $\Delta t_{\text{cycle}}(k)$, i.e. $t_\delta \leq \Delta t_{\text{cycle}}(k)$. On the other hand, one data block forwarded to the decoding buffer can complete a generation in case of $\tau > 0$. Thus, one data block is sufficient to trigger the subsequent decoding process. For this purpose, the decoding interval t_δ can take a value in the range $\frac{\Delta t_{\text{cycle}}(k)}{k} \leq t_\delta \leq \Delta t_{\text{cycle}}(k)$. Although a short decoding interval t_δ will release complete generations very rapidly, a new decoding interval can only start after a new generation has reached the decoding buffer, which in worst case can be as long as Δt_{cycle}. Thus, the longest possible decoding interval is the best estimate, and defined as $t_\delta = \Delta t_{\text{cycle}}(k)$. As a result, the decoding queue size can be calculated by combining Eq.(2.34), Eq.(2.61) and Eq.(2.62) as

$$\Omega_{\text{LNC}} = \Omega_\delta(\Delta t_{\text{cycle}}(k)) + \Omega_{\text{ini}}(\tau) = k + \tau(k, \alpha). \tag{2.63}$$

We next analyze the VOQ buffer architecture with redundancy, i.e., $n = k + r$, $r \geq 0$, and random path selection. We verify in this way that under regular operation mode, i.e., in absence of path failures,

the redundant paths not only can reduce the skew $\bar{\tau}(r)$, as discussed earlier, but also the buffer size required. Similarly to the previous analysis, derived from Eq. (2.63) with Eq. (2.49), the average decoding buffer size is given by

$$\Omega_{\mathrm{LNC}}^{\mathrm{R}}(r) = k + E\{\sum_{m=1}^{n-r} \tau_m(\alpha, r)\} = k + \sum_{m=1}^{n-r} E\{\tau_m(\alpha, r)\}, \tag{2.64}$$

where $E\{\tau_m(\alpha, r)\}$ is an expected value of the path-specific skew, defined below. To derive this value, for each path set $\mathcal{M}_n(\alpha)$, we define the difference between neighboring paths $\mathcal{P}_{l_{\nu+1}}(\alpha)$ and $\mathcal{P}_{l_\nu}(\alpha)$, as $\Delta\tau_\nu(\alpha, r) = d_{l_{\nu+1}}(\alpha) - d_{l_\nu}(\alpha)$, $\nu = 1, 2, ..., n-r-1$. Following Eq. (2.23) and using $\tau_m(\alpha, r) = d_{l_{n-r}}(\alpha) - d_{l_m}(\alpha) = \sum_{\nu=m}^{n-r-1} \Delta\tau_\nu(\alpha, r)$ the expected value of path-specific skew is

$$E\{\tau_m(\alpha, r)\} = \sum_{\nu=m}^{n-r-1} E\{\Delta\tau_\nu(\alpha, r)\}, 1 \leq m \leq n-r-1, \tag{2.65}$$

and $E\{\tau_{n-r}(\alpha, r)\} = 0$, where $E\{\Delta\tau_\nu(\alpha, r)\}$ is the expected value of the delay difference between two sequent lanes from $\mathcal{M}_n(\alpha)$. To get more insights into the general behavior of the path selection, these values are assumed to be identical, i.e. $E\{\Delta\tau_\nu(\alpha, r)\} = E\{\Delta\tau(\alpha, r)\} = \Delta\bar{\tau}$. Under this assumption, the expected value of the largest skew follows to be $E\{\tau_1(\alpha, r)\} = \bar{\tau}(r) = (n-r-1)\Delta\bar{\tau}$, which allows to derive $\Delta\bar{\tau} = \frac{\bar{\tau}(r)}{n-r-1} = \frac{\bar{\tau}(r)}{k-1}$. Furthermore, with $k = n-r$ Eq. (2.65) can be simplified to $E\{\tau_m(\alpha, r)\} = \sum_{\nu=m}^{k-1} \Delta\bar{\tau} = (k-m)\Delta\bar{\tau}$. Thus, with Eq. (2.49) an approximation for Eq. (2.64) is

$$\Omega_{\mathrm{LNC}}^{\mathrm{R}}(r) \approx k + \sum_{m=1}^{k} (k-m)\Delta\bar{\tau} = k + \bar{\tau}(r) \cdot \frac{k}{2}. \tag{2.66}$$

When no optimization is applied and data blocks are sent over any $n = k$ out of N paths, the expected value of the queue size can be derived in closed form. With the average size of the deskew buffer defined by Eq. (2.36), we can compare both buffer architectures in a

generalized statistical way. Thus let us estimate the advantage of the decoding over the deskew based buffer architecture using Eq.(2.36) and (2.66), if $r = 0$ and $\bar{\tau}(0) = \bar{\tau}$, which yields

$$\frac{\Omega^{\mathrm{R}}_{\mathrm{ML}} - \Omega^{\mathrm{R}}_{\mathrm{LNC}}(0)}{\Omega^{\mathrm{R}}_{\mathrm{ML}}} \approx \frac{\bar{\tau}}{2(\bar{\tau}+1)} \xrightarrow[\bar{\tau} \gg 1]{} \frac{1}{2}. \tag{2.67}$$

In conclusion, the decoding, i.e., VOQ-based, buffer architecture outperforms on the advantage the deskew architecture independent of the number of parallel paths used in the network, with an up to 50% improvement in buffer size in case of large skews.

Analysis of Upper and Lower Bounds of Deskew Buffer

The upper bound of the decoding buffer, $\Omega^{\mathrm{R}}_{\mathrm{up}}(r)$, $r \geq 0$, can be derived by considering all N available paths with the corresponding path delays. The worst case for the decoding queue size occurs when $k-1$ data flows with delays $d_1 \leq d_2 \leq ... \leq d_{k-1}$ from $\vec{d}$ arrive and need to be buffered at the receiver, while the k^{th} data block from each generation required for decoding start always takes a path with the largest path delay d_{N-r}, according to Eq. (2.47). This path set is denoted as $\mathcal{M}_n(\alpha_{\mathrm{up}})$ and with $\tau_m(\alpha_{\mathrm{up}}) = d_{(N-r)_k}(\alpha_{\mathrm{up}}) - d_{l_m}(\alpha_{\mathrm{up}})$, where $d_{l_m}(\alpha_{\mathrm{up}})$ can be $d_{1_1}(\alpha_{\mathrm{up}}) \leq d_{2_2}(\alpha_{\mathrm{up}}) \leq ... \leq d_{(k-1)_{k-1}}(\alpha_{\mathrm{up}}) \leq d_{(N-r)_k}(\alpha_{\mathrm{up}})$ the upper bound is given as follows

$$\Omega^{\mathrm{R}}_{\mathrm{up}}(r) = k + \sum_{m=1}^{n-r} \tau_m(\alpha_{\mathrm{up}}). \tag{2.68}$$

Next, let us establish the absolute lower and upper bounds on Eq. (2.63) for arbitrary path sets and topologies. For the upper bounds, we define that $k-1$ data flows received at the same shortest time and, thus, have $\tau_m = \tau_{\mathrm{up}}, m = 1, 2, ..., k-1$, with τ_{up} given in Eq. (2.26) and, thus, the buffer upper bound is determined as follows

$$\Omega_{\mathrm{up}} = k + (k-1) \cdot \tau_{\mathrm{up}}. \tag{2.69}$$

For lower bounds, in contrast, we define that $k-1$ paths have the same maximum delay and one path has a minimal delay, e.g.,$\tau_m = 0, m = 2, ..., k$ and $\tau_1 = \tau_{\text{up}}$. With Eq. (2.63), the lower bound is

$$\Omega_{\text{low}} = k + \tau_{\text{up}}. \tag{2.70}$$

2.5 Performance Evaluation

On an example of an Ethernet-over-optical network, we now present the numerical results and validate the same by simulations. All data blocks have the same size and LNC is implemented in a field F_{2^8}, the number of lanes is set to $k = 4$ and $k = 8$. The optical network presents a fiber-based network where a pair of nodes is interconnected with 10 parallel links. In other words, each pair of nodes is connected by a fiber link, and each fiber link carries 10 wavelengths, e.g., each at 10Gb/s. The traffic load is assumed to be 1 data block per *time unit* per lane. The normalized confidence intervals show 95% confidence and small values and are not shown for clarity.

The numerical results shown in the remainder compare five basic methods of parallel transmission implemented either in the circuit-switched network or in the packet-switched network, i.e.,

1) OPT. This method relates to optimal routing in circuit-switched networks and does not use LNC, whereby a set of optimal parallel paths with minimum skew is found. The number of paths n chosen equals the number of Ethernet lanes k, i.e., $n = k$. The deskew buffer is determined by Eq. (2.35), while the skew τ is known and constant as per Eq. (2.19).

2) MP-RND. This method relates to random routing in circuit-switched networks but does not use LNC, whereby a set of parallel randomly selected paths is utilized for the parallel transmission. The number of transmission paths n equals the number of Ethernet lanes k, i.e., $n = k$. The deskew buffer is determined by Eq. (2.39), whereby the expected skew $\bar{\tau}$ is calculated with Eq. (2.25).

3) C-OPT. That is a variation of method 1) with the difference that LNC is applied, while $n = k$ paths in the network are chosen to yield the minimum skew τ, which is known and constant (Eq. (2.19)). The decoding buffer size is found with Eq. (2.63).
4) C-RND. This method corresponds to the coded random routed parallel transmission in circuit-switched networks, whereby $n = k+r$ paths are randomly selected out of N available paths. The decoding buffer size is determined by Eq. (2.64), while the delay pattern of randomly selected paths are determined by Eqs. (2.23) and (2.48) to analyze skew with Eq. (2.49).
5) PS-RND. This method relates to random packet switching without LNC, whereby any forwarding node randomly selects ω outgoing links to send ω incoming packets in parallel. The number of outgoing links n at source equals the number of Ethernet lanes k, i.e., $n = k$. The required buffer at the receiver is determined by Eq. (2.40)and the expected skew $\bar{\tau}$ is calculated with Eq. (2.33).

2.5.1 Circuit Switched Networks

For a circuit-switched network, we analyze two scenarios. In one scenario, we assume an abstract network topology that is presented by F parallel disjoint paths between source and destination, where F can be an arbitrary number. The path delay d_l, where $1 \leq l \leq F$, is a function of path index l determined as $d_l = l\ tu$. In the second scenario, we study the known nsfNet network with 14 nodes and 21 fiber links presented in Fig. 2.7. Here, we show the results for the traffic transmission between node 0 as a sender and node 5 as a receiver, a choice of which is used entirely for illustration. The network is modeled as a directed and acyclic graph and presents a transparent fiber-based network and, thus, is able to connect 10 end-system interfaces at a time, e.g., each at 10Gb/s. Thus, it is possible to build 11 different fiber paths, illustrated in Table 2.1, and on these fiber paths, 10 wavelengths paths can be set up in parallel and have

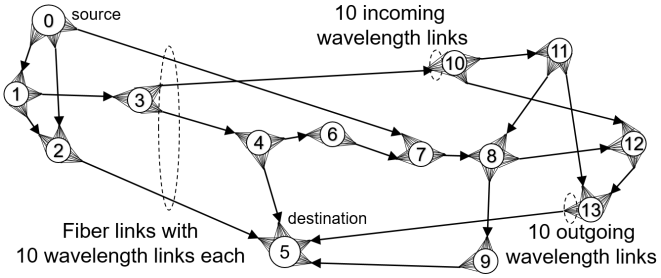


Figure 2.7: nsfNet topology [1].

l	$\mathcal{P}_l$	l	$\mathcal{P}_l$
1	0-2-5	7	0-1-3-10-11-13-5
2	0-1-2-5	8	0-1-3-10-11-8-9-5
3	0-7-8-9-5	9	0-1-3-10-11-8-12-13-5
4	0-1-3-4-5	10	0-1-3-4-6-7-8-9-5
5	0-7-8-12-13-5	11	0-1-3-4-6-7-8-12-13-5
6	0-1-3-10-12-13-5		

Table 2.1: Optical fiber paths between nodes 0 and 5 [1].

the same end-to-end delay. Each link in nsfNet exhibits the same delay of $1tu$. For instance, on path 2, there are 3 fiber hops and all 10 wavelengths within this fiber path have the same end-to-end delay of $3tu$. From the perspective of skew, however, only paths with a different number of hops are interesting, as this scenario creates the greatest spreading of skews. Therefore, any optical wavelength can be utilized on any out of 11 existing fiber paths. Thus, we consider in total only $F' = 11$ wavelength disjoint paths referring to as $F \equiv F'$ below. The resulting delay vector is $\vec{d} = (2, 3, 4, 4, 5, 6, 6, 7, 8, 8, 9)$.

In both scenarios, our analysis is based on a fixed set of available paths N with reduced delay vector $\vec{d}$. For example, with $N < F$, there are $a_{\mathcal{F}} = C_{F,N}$ possible path combinations $\mathcal{F}_{N_\nu}, \nu = 1, ..., a_{\mathcal{F}}$, to select N out of F paths, each with different $\vec{d}$. The presented analytical and simulation results are averaged over all $a_{\mathcal{F}}$ combina-

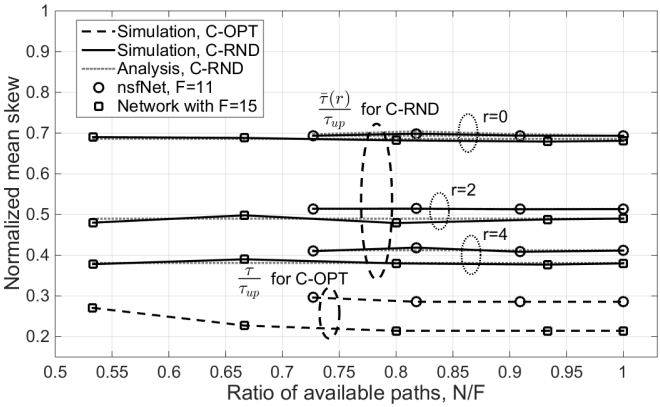


Figure 2.8: Normalized mean skew ($k = 4$) [1].

tions. Let us denote $f_\nu(n, N), \nu = 1, ..., a_\mathcal{F}$, as results determined for specific combinations $\mathcal{F}_{N_\nu}$, then all results shown for N/F can be expressed as $E(f(n, N)) = \frac{1}{a_\mathcal{F}} \sum_{\nu=1}^{a_\mathcal{F}} f_\nu(n, N)$.

Evaluation of Skew

First, we assume that the network is lossless and fault-free and analyze the skew as a function of the total number of paths available, N out of F possible paths. The mean skew is normalized by the maximum skew $\tau_{\text{up}}^F = (d_F - d_1)tu$. In the abstract network we assume $\tau_{\text{up}}^{15} = 14tu$ for $F = 15$, and $\tau_{\text{up}} = 7$ for nsfNet.

Fig. 2.8 analyses the same scenarios with redundancy. As expected, the normalized mean skew decreases and nears the optimal value with a growing number of redundancies and reaches around 39% and 41% of the maximum skew τ_{up} of an abstract network with $F = 15$ and nsfNet, respectively. Here, the theoretical results were calculated with Eqs. (2.26) and (2.49), and match simulations.

Next, we assumed that all wavelength paths in the network are

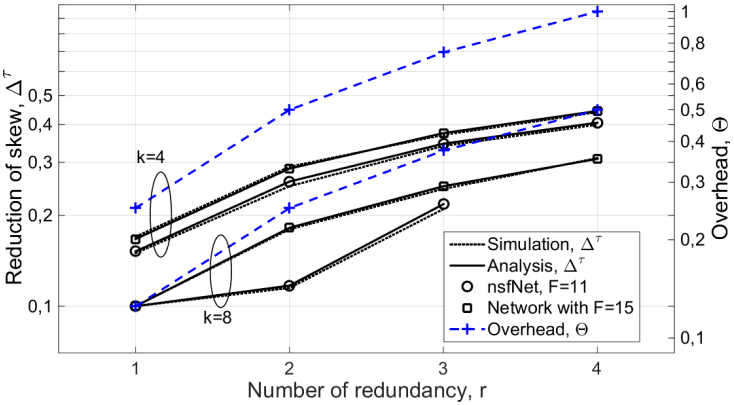


Figure 2.9: Reduction of skew (C-RND) [1].

available, i.e., $N = F$, and compared nsfNet ($F = 11$) and the abstract network with $F = 15$.

Fig. 2.9 shows the reduction of skew and the corresponding transmission overhead (Eq. (2.55) and Eq. (2.44)) as a function of the number of lanes k and redundancy r. The simulations and analysis show that an increased number of redundancies r reduces the skew, of about 30% for $r = 3$, and rapidly increases the overhead.

Fig. 2.10 shows the occurrence rate of a maximum skew τ_{up} in case of C-RND (abstract network with $N = F$ available paths) and $D_{\min} = D_{\max} = 1$. The analytical results are based on Eqs. (2.29) and (2.53), while τ_{up} was defined according to Eq. (2.26). The occurrence rate of τ_{up} decreases with increasing number of redundancies r and, at the same time, increases with increasing k, e.g., $P_{\text{up}} \approx 0.057$ for $k = 4$ and $P_{\text{up}} \approx 0.27$ for $k = 8$ and $r = 0$ in a network with $N = 15$ available paths. The occurrence probability of a maximum skew $P_{\text{up}}(0)$ is reduced from around 5.7% to around 1.3%, when the number of available paths is doubled, i.e., changed from $N = 15$ to $N = 30$, respectively, while transmission was established

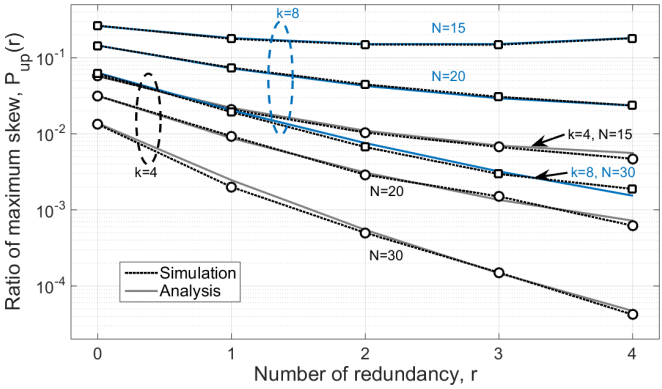


Figure 2.10: The occurrence rate of a maximum skew (C-RND) [1].

over $n = k = 4$ parallel paths. The transmission over $n = k = 8$ parallel paths in the same scenario result in a change of $P_{\mathrm{up}}(0)$ from around 27% to around 6%. That is because an increase in N results in increase in number of equally probable paths combinations.

Evaluation of De-Skew Buffer

We next evaluate the receiver buffer size in the case of parallel transmission over ether optimal or random routing paths.

Fig. 2.11 illustrates that C-RND with redundancy can outperform the methods with path optimizations (C-OPT and OPT) in terms of the buffer size, even for a higher path blocking probability. The buffer requirement for C-RND is nearly constant for all values of the path blocking probability P_{B} and can be reduced by up to 43% of upper bound $\Omega^{\mathrm{R}}_{\mathrm{up}}(0)$ defined with Eq. (2.68) by increasing the number of redundant paths up to $r = 4$. In contrast, the buffer requirement in case of C-OPT and OPT increases with increasing P_{B} from 33% to 60% and 37% to 85%, respectively.

Finally, we study the impact of path failures on decoding buffer

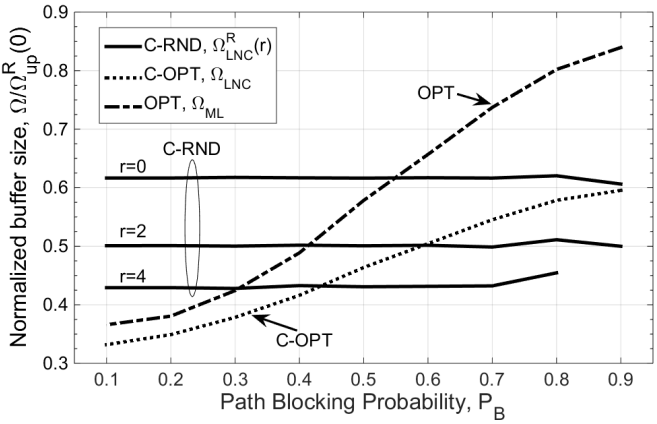


Figure 2.11: Buffer size vs. path blocking probability ($k = 4$) [1].

for C-RND. The results in Fig. 2.12 are observed for nsfNet with $N = F = 11$ and $N = 8$ available paths. The generation size is set being $k = 4$, and the amount of redundancy is varied from $r = 0$ to $r = 4$, whereby $f \leq r$ paths failures were allowed. As a worst case scenario, we show the normalized expected queue size of decoding buffer $\frac{\Omega^{R}_{LNC}(r)}{\Omega^{R}_{up}(0)}$, when exactly $f = r$ paths fail in any randomly selected path combination. The expected skews $\bar{\tau}_2(r, f)$ and related queue size $\Omega^{R}_{LNC}(r)$ were calculated by Eqs. (2.60) and (2.66), respectively. Since each possible subset of the path is affected by all failures, there is no remaining redundancy. As a consequence, the receiver queue size is independent of the N/F ratio, constant for any $f = r$ and equal to the required decoding buffer in case of transmission without redundancy over a lossless optical network, i.e., $f = r = 0$. Thus, C-RND always compensates for the worst case failure scenario. That is compared to the practical case in which errors may not affect each randomly selected path set. The results are calculated with Eqs. (2.57) (2.58) and (2.66). As expected, the buffer

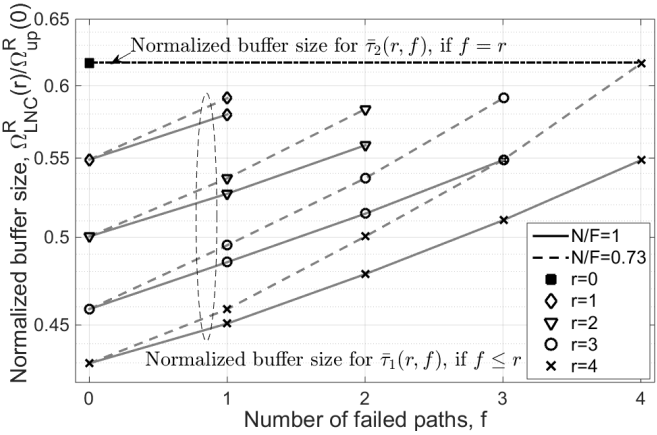


Figure 2.12: Normalized queue size ($k = 4$) [1].

size increases with the number of failed paths f and the decreasing number of available paths N and redundancy r. If compared, however, to the scenarios without failures and redundancy (equal to the worst case scenario), the randomized selection of routing paths requires at average smaller buffer size because network failures only partially affect the chosen parallel paths.

2.5.2 Packet Switched Networks

Next, we numerically evaluate the performance of random optical packet switching (PS-RND) and compare them with random circuit switching (MP-RND). All results for skew and queue size are normalized by the upper bound of skew defined with Eq.(2.26) and by the upper bound of queue size of PS-RND defined with Eq.(2.40) in the case of 0.5075 available network capacity. For the simulations and analysis, we use nsfNet network shown in Fig. 2.7, where link direction were defined as $\{0-1, 0-2, 0-7, 1-2, 1-3, 2-5, 3-10, 3-4, 4-5, 4-6, 5-9, 5-13, 6-7, 7-8, 8-11, 8-12, 9-8, 10-11, 10-12, 11-13, 12-13\}$. In the case of PS-RND, each

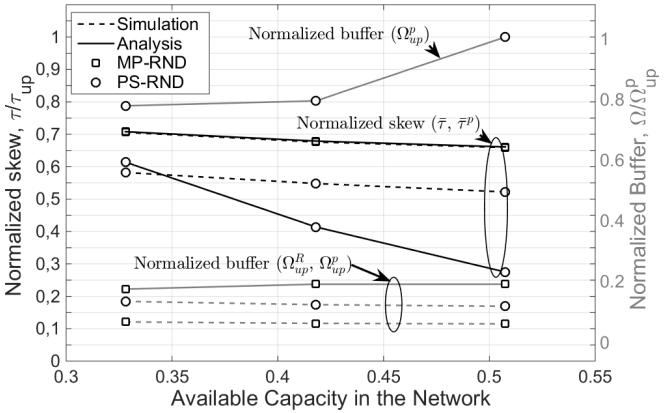


Figure 2.13: Normalized skew and buffer size (n=4) [3].

set of optical packets is sent in parallel on a different set of $n = k$ randomly selected links. Moreover, each optical packet experiences the same node-to-node delay of 5 tu on each link, i.e., $\varepsilon = 5tu$.

In the first set of results, we assume no wavelength conversion and no blocking. Thus, all-optical packets could be switched to the same output wavelength in any forwarding node. Fig. 2.13 shows a comparison of skew and buffer requirements for MP-RND and PS-RND methods as a function of available capacity in the network, for $n = 4$. The mean skew and the upper bounds of buffer size were analytically defined with Eqs.(2.25), (2.33) and Eqs. (2.39), (2.40), respectively. Generally, the mean skew decreases with increasing available network capacity and is up to 14% larger in the case of MP-RND as compared to PS-RND. The better performance of PS-RND can be explained by the fact that the packet-switched networks provide more available partly disjoint paths, i.e., a higher number of shorter paths. In contrast, the amount of paths used for MP-RND is reduced due to the requirement of disjointedness. In the case of MP-RND, the simulation results overlap with analytical results for skew. In contrast, there is a deviation between simulation and

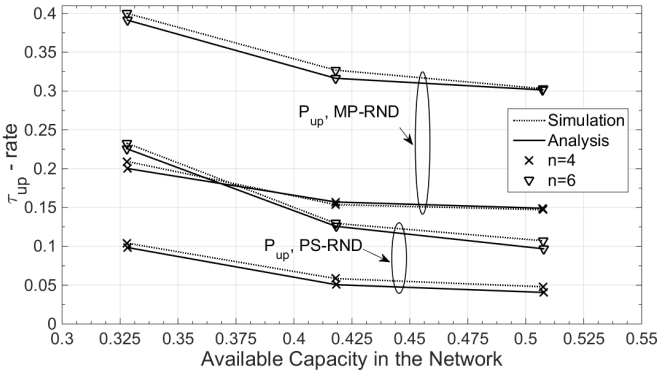


Figure 2.14: Occurrence rate of maximum skew τ_{up} [3].

analytical results for PS-RND, which increases with the increasing available network capacity, i.e. number of available parallel paths to each forwarding node and to the destination. That is because the analytical path occurrence probability is calculated with Eq. (2.8) based on link probabilities defined by Eq. (2.13). As a result, in analysis, the shorter paths have a higher occurrence probability, whereby the link occurrence probability is constant. Here, we do not consider special scenarios in forwarding nodes, where multiple, i.e., $\omega > 1$, packets arrive at the same time and forwarded in parallel, i.e. forwarding node v selects randomly a set of ω outgoing links with changing probability $\phi_{e_j}(\omega) = \sqrt[\omega]{\frac{1}{C_{|\mathcal{E}_{out}(v)|,\omega}}}$ each. The simulation results show that the resulting queue size at the receiver for PS-RND is only around 6% larger than for MP-RND. That is due to the fact that, in contrast to MP-RND, PS-RND spreads sent packets over $\omega >> n$ incoming links at the destination node and increases the number of packets arrived per time unit. The theoretical upper bounds are not exceeded, because analysis assumed that the longest path is always used for both MP-RND and PS-RND, which will not be the case in a real network. Generally, the skew and the buffer size will increase with an increasing number of parallel lanes $k = n$ utilized in the source.

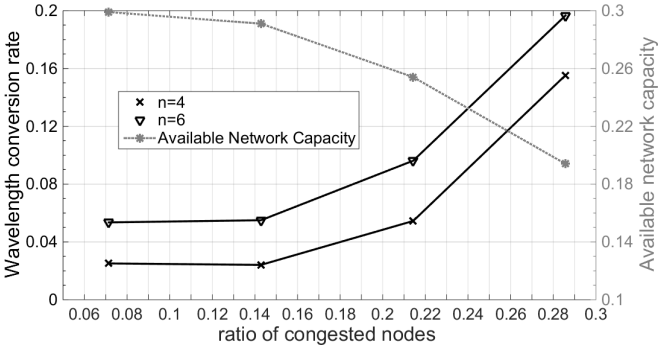


Figure 2.15: Mean wavelength conversion ratio (PS-RND) [3].

Fig. 2.14 shows the occurrence rate of the maximum skew τ_{up}, which can reach up to 10% in a heavily loaded packet-switched network with $n = 4$, while the MP-RND method often selects (around 21%) the shortest and the longest paths for transmission. Here, simulation results are very close to the analytical results. The increase in n increases the occurrence rate of the maximum skew, e.g. doubled for $n = 6$ as compared to $n = 4$.

Fig.2.15 shows the simulation results for the wavelength conversion rate, which would be required in OPS networks without a sufficient amount of capacity to ensure wavelength continuity constraint. Thus, the conversion rate is defined as a fraction of a mean number of optical packets switched to outgoing link with converted wavelength to the number of all transmitted packets. The results show that only 3% of optical packets have to be wavelength switched, if there are around 7% of congested nodes in the network, which provides 30% available capacity. As expected, with an increasing number of congested nodes, increasing n, and low available network capacity, the conversion rate increases, up to 20%.

2.6 Summary

This chapter studied parallel transmission systems with coding and random routing in circuit-switched and packet-switched networks. We provided a novel analysis based on combinatorics and compared optimal and random routing regarding skew and deskew buffer. The results showed that random circuit switching requires less buffer at the receiver, while random packet switching provides better performance in regard to skew. The performance of parallel transmission over circuit-switched networks in combination with random routing and erasure coding was analyzed as well. The expected values as well as the upper and lower bounds of the skew showed their independence of the number of available paths in the network, which is an interesting result. In the case of path failures, the parallel transmission with coding can provide transmission reliability and had the mean deskew buffer, which is smaller as compared to conventional systems. The results, in fact, showed a great promise of parallel network systems in general and applications of erasure coding. With a proper set of design parameters, the coding redundancy reduces the mean and the upper bound of skew and deskew buffer size, which is relevant in the case of random routing.

2.7 List of Symbols

k	number of virtual Ethernet lanes related to generation size;
r	number of redundant transmission paths related to coding redundancy;
n	number of parallel transmission paths related to coded generation size, i.e., $n = k + r$;
$G(V, E)$	Directed and acyclic graph of the network;
V	Set of nodes in the network;
E	Set of links in the network;
$\mathcal{E}_{in}(v)$	Set of incoming links of an arbitrary node v;
$\mathcal{E}_{out}(v)$	Set of outgoing links of an arbitrary node v;
$y_{in}(v)$	number of incoming links in node v in packet-switched network;
$y_{out}(v)$	number of outgoing links in node v in packet-switched network;
$G_{\mathcal{P}}$	ordered set of available parallel paths between source and destination;
$\mathfrak{N}$	number of available parallel paths between source and destination;

$\mathcal{P}_l$	certain available path from set $G_{\mathcal{P}}$, where $1 \leq l \leq \mathfrak{N}$;
$\vec{d}$	vector of length $\mathfrak{N}$ containing end-to-end delays of available paths;
d_l	end-to-end delay of path $\mathcal{P}_l$ collected in delay vector $\vec{d}$;
$D_{\max}$	number of components d_l in delay vector $\vec{d}$, which are equal to component d_N;
$D_{\min}$	number of components d_l in delay vector $\vec{d}$, which are equal to component d_1;
$\mathcal{T}$	flow completion time in general;
τ	value of skew in general;
$\bar{\tau}$	mean value of skew;
τ_{up}	maximum value of skew;
P_{up}	occurrence probability of maximum skew τ_{up};
$\mathcal{M}$	collection of all possible path combinations $\mathcal{M}_n(\alpha)$ of n paths randomly selected for transmission;
$\mathcal{M}_n(\alpha)$	set of n paths from $G_{\mathcal{P}}$, where $\mathcal{M}_n(\alpha) \in \mathcal{M}$;
$a_{\mathcal{M}}$	number of possible combinations of n paths over $\mathfrak{N}$ available paths;

Ξ	collection of all possible combinations $\Xi_f(\beta)$ of f paths randomly selected and failed during transmission, where $1 \leq \beta \leq C_{\mathfrak{N},f}$;
$\Xi_f(\beta)$	set of f paths randomly selected from $G_{\mathcal{P}}$ and failed during transmission, where $\Xi_f(\beta) \in \Xi$ and $1 \leq \beta \leq C_{\mathfrak{N},f}$;
$\mathcal{P}_{l_m}$	certain available path, which belongs to set $G_{\mathcal{P}}$, where $1 \leq l \leq \mathfrak{N}$, and set $\mathcal{M}_n(\alpha)$, where $1 \leq m \leq n$ and $1 \leq \alpha \leq a_{\mathcal{M}}$;
d_{l_m}	end-to-end delay of path $\mathcal{P}_{l_m}$;
F	total number of link disjoint paths in circuit-switched network;
N	number of available link disjoint paths in circuit-switched network;
P_{setup}	path setup probability in circuit-switched networks;
P_{B}	probability of path blocking in circuit-switched networks;
$B(n)$	probability of request blocking in circuit-switched networks;
Ψ	number of available partly disjoint paths in packet-switched network;
$\mathfrak{y}_l$	number of utilized links to compose a path $\mathcal{P}_l$;

ρ_{lh}	h^{th} link utilized to compose a path $\mathcal{P}_l$, $1 \leq h \leq \mathfrak{y}_l$;
ϕ_{lh}	probability to utilize link ρ_{lh} of path $\mathcal{P}_l$ in packet-switched network;
ϕ_{e_j}	probability to utilize link e_j in node v, where $e_j \in \mathcal{E}_{out}(v)$ and $1 \leq j \leq y_{out}(v)$, in packet-switched network;
v_{lq}	q^{th} node utilized to compose a path $\mathcal{P}_l$, $1 \leq q \leq \mathfrak{y}_l - 1$;
ε_{lh}	node-to-node delay of link ρ_{lh} on the certain path $\mathcal{P}_l$;
ω	number of packets arrived simultaneously at an intermediate node;
$\mathcal{M}_\omega(\alpha)$	set of ω paths out of Ψ available paths;
$P_B(v)$	probability of packet dropping or buffering in node v;
$\mathfrak{G}$	ratio of the congested nodes in packet switched network;
ϱ_q	buffering and processing delay in intermediate node v_{lq} on path $\mathcal{P}_l$ in packet switched network;
M	number of data blocks in the source data;
L_M	amount of bits in each data block;
C	bit rate of transmission system;

Ω	size of deskew buffer in general;
F_{2^b}	finite field with a symbol size of b bits;
$p_e^{'}$	bit error rate in the physical layer;
R	code rate in general;
h_{min}	minimal Hamming distance;
t_e	number of erroneous bits detected with FEC;
t_k	number of bits corrected with FEC;
$\boldsymbol{\Theta_t}$	transmission overhead due to coding redundancy;
Θ	total transmission overhead due to coding redundancy;
Φ	information flow;
$\boldsymbol{\Delta}^{\max}$	reduction of maximum skew;
$\boldsymbol{\Delta}^{\tau}$	reduction of mean skew.

3

A Combinatorial Reliability Analysis of Advanced Network Services

3.1 Introduction

The service reliability in advanced network systems can be considered as a probability that the system will successfully complete the processing of the requested service. As such, a reliability analysis in system engineering is critically important to facilitate the design of reliable systems by evaluating system robustness in case of any system failures. The failures of software and hardware components can be caused by a misconfiguration, overload or security attack. To address issues of data integrity and reliability, in this chapter, we design, analyze and evaluate advanced network services for reliable data transmission and reliable network function chaining. Diversity, coding and over-provisioning are selected as the main tools. Thus, here, parallel transmission, erasure coding and different types of redundancy, e.g., coding, path, system components, are utilized to design reliable network services.

Since the quality of any wireless transmission strongly depends on the transmission environment, the changing weather condition and other atmospheric effects, in this chapter, we investigate a reliable wireless transmission as a service on an example of Free Space Optics

(FSO). FSO refers to wireless point-to-point communication, based on a line-of-sight optical technology that allows full-duplex communication at the speed of light. The FSO communication system presents several advantages with regard to power consumption, electromagnetic interference and the high bandwidth of up to 100Gbps. The main disadvantage, however, is their sensitivity to atmospheric phenomena such as fog decreasing the quality of FSO signal, i.e., transmission reliability [59, 60, 61, 62]. To address the issue of signal quality affected by atmospheric effects, it was proposed to use channel coding as well as data transmission over multiple diverse channels, [63, 64, 65, 66, 67, 68, 69]. Hence, additional supporting channels, e.g., radio-frequency (RF) or FSO channels can be used. For instance, a system with multiple parallel FSO channels carrying the same data has been proposed [64, 67, 70]. It was shown that hybrid use of FSO and RF significantly improves the reliability of data transmission in foggy and rainy conditions, because FSO channels are attenuated by fog, while the RF channel is, in contrast, mainly attenuated by rain[69]. In this chapter, however, the use of diversity, coding and over-provisioning is entirely different due to the parallel transmission concept applied.

The second type of reliable services considered in this chapter is network services provided by Service Function Chains (SFC), i.e., a composition of Virtual Network Functions (VNFs) distributed over Inter- and Intra-DCN. Since Equal Cost Multipath Protocol (ECMP) is the standard protocol in DCNs, the traffic splitting into sub-flows and their parallel transmission, i.e., path diversity, are valid [17, 18, 19]. However, the resulting sub-flows will require separated and independent processing to maintain diversity during service run time and, thus, multiple replicas of the same SFC, i.e., diversity of DCN components such as servers and VNFs. Without over-provisioning, the reliability of an SFC is critical as a failure of any VNF disrupts the entire SFC causing service interruptions. To

achieve the required level of SFC reliability, the most common failure protection technique is the backup VNF protection [71, 72, 73, 74]. However, to be effectively used, the backup protection needs fast failure detection and rapid reaction to the detected failures to minimize the traffic losses. As a result, the backup protection requires fast VNF migration and traffic redirection to the backup VNFs, which are challenging tasks [26, 75, 76, 77, 78, 79]. As diversity and overprovisioning seem to be not sufficient to provide effective VNF failure protection, there is a need to investigate the impact of the third tool, i.e., coding, on the SFC reliability and failure management. This chapter provides a novel VNF protection with coding to increase SFC reliability and simplify the handling of VNF failures.

Another challenge addressed in this chapter is assessing SFC reliability and dimensioning of backup resources as a function of the VNF placement strategy. Generally, analysis of SFC reliability is a complex problem as it needs consideration of co-location and sharing of hardware components, heterogeneity of software and hardware and complex interdependency between them [27]. Previous analytical approaches such as in [80, 81, 82] tried to limit the complexity of SFC reliability evaluation. To this end, some of them considered the SFC provisioning with serial traffic flows, whereby the reliability of the individual DCN components was taken into account separately. Others supposed failures of multiple components but used assumptions such as disjointness and no interoperability between them. However, failures of individual components involved in SFC are not independent. Ignoring of components' interdependency during reliability prediction can be fatal as any single failure of any component can become a cause of other component failures and service interruption despite failure protection mechanisms utilized. The previous reliability analysis models cannot be directly applied to SFC because they do not consider aspects of system component sharing, heterogeneity of system components, their interdependency in the case of

failures and complex failure propagation. That makes the reliability analysis of SFCs with any protection method a challenge, which has not been addressed yet analytically.

This chapter first presents a new approach to exploit channel diversity and combine redundancy of parallel transmission with the redundancy of an adaptive channel coding to increase FSO transmission reliability. To ensure channel diversity the parallel transmission can be realized using either multiple FSO or both RF and FSO channels. However, the main challenge is to adapt channel coding parameters to the channel conditions. To achieve an adaptive channel coding the erasure coding is suitable due to its flexibility and ability for erasure correction. Among many candidates, we chose the random Linear Network Coding (LNC) because it can provide an arbitrary length of coded data blocks without length limit. Thus, with LNC, the encoded block length, the number of source bits encoded and the amount of redundancy for error correction can be variable. Moreover, a simple but effective algorithm for an adaption of coding parameters is provided and verified. Finally, this chapter presents a novel analytical model and analysis of the SFC reliability in DCNs that deploys flow and SFC parallelism, backup, and coding failure protections. We investigate how instead of backup protection, a *systematic erasure coding* can improve service reliability in DCNs while alleviating or even eliminating the challenges associated with VNF failure detection, VNF migration, and traffic redirection. The provided novel generalized reliability analysis based on combinatorial analysis and a reduced binomial theorem enables evaluation of the SFC reliability as a function of VNF placement strategies in generic inter- and intra- DCNs. To this end, failures of data centers, racks, servers, VMs and path segments, and besides system component sharing, heterogeneity, their interdependencies and failure propagation are taken into account in the analysis.

3.2 Supporting Publications

1. A. Engelmann and A. Jukan, “A Combinatorial Reliability Analysis of Generic Service Function Chains in Data Center Networks,” in *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, under submission.

2. A. Engelmann, W. Bziuk and A. Jukan, ”Bounding Reliability in Service Function Chaining,” 2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 2020, pp. 413-418, doi: 10.23919/MIPRO48935.2020.9245235.

3. A. Engelmann, A. Jukan and R. Pries, ”On Coding for Reliable VNF Chaining in DCNs,” 2019 15th International Conference on the Design of Reliable Communication Networks (DRCN), Coimbra, Portugal, 2019, pp. 83-90. doi: 10.1109/DRCN.2019.8713668

4. A. Engelmann and A. Jukan, ”A Reliability Study of Parallelized VNF Chaining,” 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, 2018, pp. 1-6. doi: 10.1109/ICC.2018.8422595

5. A. Engelmann and A. Jukan, ”Serial, parallel or hybrid: Towards a highly reliable transmission in RF/FSO network systems,” 2015 IEEE International Conference on Communications (ICC), London, 2015, pp. 6181-6186. doi: 10.1109/ICC.2015.7249308

3.3 Reliable Parallel Transmission as a Service

In this section, a reliable parallel transmission over FSO channels is considered as a service. To ensure transmission reliability, the adaptive channel coding is applied to the data prior to FSO transmission. The list of used symbols is summarized in Sec. 3.3.4.

3.3.1 Adaptive Coding Approach

The network traffic is represented as a sequence of symbols of the same size b, i.e., the same number of bits per symbol. Every k source symbols referred to as a *generation* are encoded into n symbols, $n > k$, which build an outgoing data block of size n. Then, any encoded data block represents a coded generation. The additional $r = n - k$ symbols are referred to as *redundancy*. These redundant symbols allow us to recover up to r corrupted symbols from the same generation at the receiver. The n encoded symbols are sent to the receiver, whereby any k out of n received symbols are sufficient for successful decoding, i.e., reliable parallel transmission. During transmission, any symbol can be affected by symbol errors, whereby the symbol error rate (SER) is denoted as P. To evaluate coding performance, the code rate is defined as $R = k/n$ and a relationship between the code rate and information rate is defined as

$$\mathcal{I} = C \cdot R, \tag{3.1}$$

where C is a total transmission rate over all transmission channels.

Fig 3.1 shows the proposed transceiver model on the example of the FSO and RF system, where some interfaces represent FSO and some RF channels. The transceiver includes up to N input (Rx) and N output (Tx) interfaces with transmission rate C_i each, whereby the total transmission rate C is a sum of transmission rates C_i of each channel i, $C = \sum_{i=1}^{N} C_i$. The communication is full-duplex, whereby the upstream and downstream from FSO channels have the same

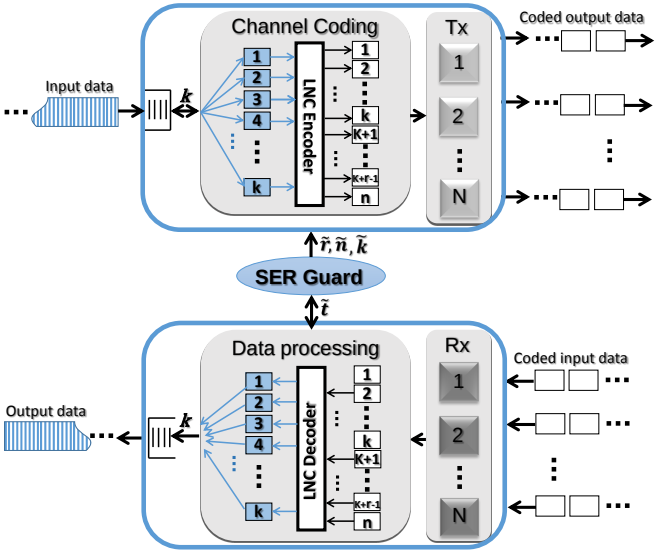


Figure 3.1: Transceiver implementation with LNC [4].

channel state, e.g., affected by the same atmospheric effects. The latter is a necessary assumption as the adaptive feature of channel coding is realized in the SER Guard (SERG) unit. Thus, the coding parameters n, k and r are adaptively defined by the SERG. The source symbol stream is first buffered into the input buffer to collect k symbols for encoding. After encoding, n symbols are distributed over at most n channels to be sent to the destination.

SERG analyses the information from the data processing unit to determine the expected number of erroneous symbols, i.e., $t^{(\nu+1)} = \tilde{t}^{(\nu)} + \delta t^{(\nu)}$, per data block, where $\tilde{t}^{(\nu)}$ is the number of erroneous symbols in the ν^{th} decoded block and $\delta t^{(\nu)}$ is a mean value of symbol error variation defined as follows

$$\delta t^{(\nu)} = \left\lceil \frac{\delta t^{(\nu-1)} + (\tilde{t}^{(\nu)} - \tilde{t}^{(\nu-1)})}{2} \right\rceil, \text{ if } \tilde{t}^{(\nu)} > \tilde{t}^{(\nu-1)}. \tag{3.2}$$

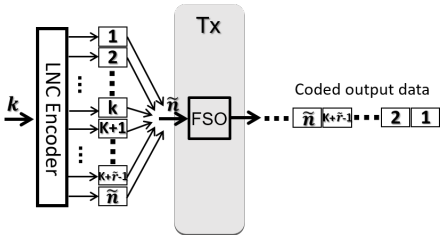
When SER goes down, i.e., $\tilde{t}^{(\nu)} < \tilde{t}^{(\nu-1)}$, the symbol error variation

$\delta t^{(\nu)}$ should not be changed immediately, i.e. $\delta t^{(\nu)} = \delta t^{(\nu-1)}$, but after an observation period. When, the number of erroneous symbols are constant during pre-defined time period, the error variation can be set to a default value, e.g., $\delta t^{(\nu)} = 0$.

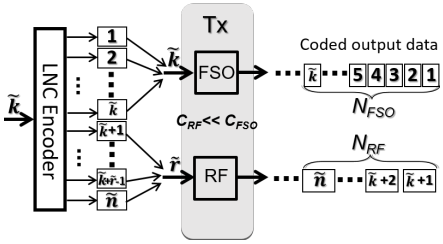
Generally, the channel quality analysis performed by SERG can follow an algorithm. However, here, the assumption is that after the channel condition is evaluated, the SERG determines the relevant transmission parameters, such as block length $\tilde{n}$, number of symbols per generation, i.e., $\tilde{k}$, and required number of redundancies $\tilde{r}$. The channel coding unit uses the parameters defined by SERG to implement channel encoding of $\tilde{k} = \tilde{n} - \tilde{r}$ symbols. For example, assume the channel coding parameters set as $n = 10$, $k = 8$ and $r = 2$ and the symbol error parameter in SERG determined as $t^{(\nu-1)} = 2$, while there was a $\tilde{t}^{(\nu-1)} = 1$ erroneous symbol in the last $(\nu-1)^{th}$ decoded generation and the error variation was calculated with Eq. (3.2) as $\delta t^{(\nu-1)} = 1$. Let us assume that there are $\tilde{t}^{(\nu)} = 3$ erroneous symbols in newly decoded generation, implying an increased number of erroneous symbols. Consequently, the new mean error variation is $\delta t^{(\nu)} = \left\lceil \frac{1+(3-1)}{2} \right\rceil = 2$ and $t^{(\nu)} = 3 + 2 = 5$. As a result, the new coding parameters are set to $n = 10$, $k = 5$ and $r = 5$, if n is fixed.

3.3.2 Design and Analysis of Reliable Transmission

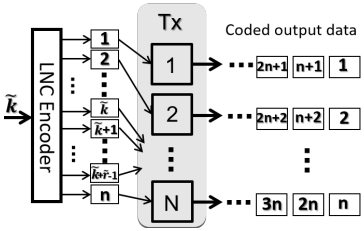
To ensure reliable transmission, there are three options to adapt the coding parameters to the channel conditions, i.e., to SER. The first option is to keep the number of source symbols k constant, and to change the number of redundant symbols, i.e., $\tilde{r}$. This is a typical *serial transmission* scenario over one transmission channel (Fig 3.2a). The second option is to keep the encoded block length n constant and decrease the number of source symbols $\tilde{k}$ per encoded block (Fig 3.2c). In this case, a *parallel transmission* can be used, where n symbols are distributed over n parallel paths. There is also the third option, which we refer to as *hybrid* as it is best suitable for



(a) Reliable Serial Transmission.



(b) Reliable Hybrid RF/FSO Transmission.



(c) Reliable Parallel Transmission.

Figure 3.2: Reliable transmission of n encoded symbols [4].

RF/FSO scenarios (Fig 3.2b). Here, the number of redundant and source symbols per block, i.e. $\tilde{r}$ and $\tilde{k}$, as well as the block length, $\tilde{n} = \tilde{k}+\tilde{r}$, are variable and depend on SER and the transmission rate ratio of both RF and FSO channels ($\frac{C_{RF}}{C_{FSO}} \geq SER$). Since k source symbols are encoded into a data block of $\tilde{n}$ symbols which is sent toward the destination, any out of $\tilde{n}$ symbols in a data block can be

corrupted resulting in the expected number of erroneous symbols:

$$t = \lceil \tilde{n} \cdot P \rceil + \delta t. \tag{3.3}$$

Thus, the number of required redundancy is $\tilde{r} = t = \lceil \tilde{n} \cdot P \rceil + \delta t$.

Reliable Serial Transmission

For the reliable serial transmission (Fig 3.2a), the block length $\tilde{n}$ can take an arbitrary value according to the channel state, whereby the number of source symbols k in each block is fixed and constant. The resulting size of encoded data block $\tilde{n}$ depends on SER, the number of source symbols k and redundant symbols $\tilde{r}$, $\tilde{n} = k + \tilde{r}$. Thus, the required number of redundant symbols can be calculated as

$$\tilde{r} = \left\lceil \frac{k \cdot P + \delta t}{1 - P} \right\rceil. \tag{3.4}$$

As a result, the expected code rate of reliable serial transmission is

$$R = \frac{k}{k + \left\lceil \frac{k \cdot P + \delta t}{1 - P} \right\rceil}, \tag{3.5}$$

while the maximal possible code rate for $P < 1$ is given by

$$R_{max} = \lim_{k \to \infty} \frac{k}{k + \left\lceil \frac{k \cdot P + \delta t}{1 - P} \right\rceil} = 1 - P. \tag{3.6}$$

Since the reliable serial transmission is implemented over one channel with transmission rate C_i, the information rate is determined with Eq. (3.1) as $\mathcal{I} = C_i \cdot \frac{k}{k + \left\lceil \frac{k \cdot P + \delta t}{1 - P} \right\rceil}$.

Reliable Hybrid Transmission

The proposed reliable hybrid communication system includes one FSO and one RF interface (Fig 3.2b), i.e., uses $N = 2$ channels. However, only the FSO channel is significantly affected by atmospheric effects and characterized by SER P. This assumption is

valid, as FSO and RF signals have different sources of interference and random LNC complements standard Forward Error Correction (FEC) mechanisms when it either fails or yields insufficient performance. Thus, the low bit error rate of the RF channel can be generally compensated by FEC in the considered scenario.

In the hybrid system, N_{RF} out of $\tilde{n}$ encoded symbols are sent over the RF channel. Thus, $N_{FSO} = \tilde{n} - N_{RF}$ symbols are sent over the FSO channel and can be corrupted with a given SER value P. Moreover, the FSO channel has a higher transmission rate C_{FSO} than the transmission rate of RF channel C_{RF}. As we assumed that all symbols have the same size of b bits, the transmission delay of RF channel, $\tau_{RF} = \frac{b}{C_{RF}}$, is larger than the transmission delay of FSO channel $\tau_{FSO} = \frac{b}{C_{FSO}}$. The propagation delays of both FSO and RF channels are, however, assumed as equal. To limit the buffer size at sender and receiver to a minimum, the symbols from an RF channel must arrive at the receiver at the latest, when the last N_{FSO}-th symbol from the FSO channel is received. Thus, the sender may send symbols from the same generation during period $\tau_{send} = N_{FSO} \cdot \tau_{FSO} = N_{RF} \cdot \tau_{RF}$ and the number of symbols to be sent over noisy FSO channel is defined as follows

$$N_{FSO} = \left\lceil \frac{N_{RF} \cdot \tau_{RF}}{\tau_{FSO}} \right\rceil = \left\lceil \frac{N_{RF} \cdot C_{FSO}}{C_{RF}} \right\rceil . \tag{3.7}$$

For a reliable hybrid transmission, it is sufficient to send $N_{RF} = \tilde{r}$ redundant symbols over the RF channel, while other $N_{FSO} = \tilde{k}$ symbols from the same generation are transmitted over the high-speed, but noisy FSO channel. However, any out of N_{FSO} symbols can be corrupted during transmission and needs to be replaced by at most N_{RF} redundant symbols from the RF channel. Thus, the number of redundant symbols sent over the RF channel can be determined as

$$N_{RF} = \lceil N_{FSO} \cdot P \rceil + \delta t. \tag{3.8}$$

With Eq. (3.7), the relation between number of sent symbols and required transmission rate of both channels is described by $\frac{N_{RF}}{N_{FSO}} \leq$

$\frac{C_{RF}}{C_{FSO}}$. At the same time, the relation between SER and the amount of sent symbols is defined by Eq. (3.8) as $P \leq \frac{N_{RF}-\delta t}{N_{FSO}}$. As a result, the reliable transmission can be achieved, if and only if

$$P \leq \frac{N_{RF} - \delta t}{N_{FSO}} \leq \frac{N_{RF}}{N_{FSO}} \leq \frac{C_{RF}}{C_{FSO}}, \tag{3.9}$$

while the maximal code rate and maximal channel utilization are achievable if all terms in Eq. (3.9) are equal. Thus, the code rate is

$$R = \frac{\left\lceil \frac{N_{RF} \cdot C_{FSO}}{C_{RF}} \right\rceil}{N_{RF} + \left\lceil \frac{N_{RF} \cdot C_{FSO}}{C_{RF}} \right\rceil}, \tag{3.10}$$

which can be simplified with Eq. (3.9) as follows

$$R \approx \frac{C_{FSO}}{C_{FSO} + C_{RF}} = \frac{1}{1+P}. \tag{3.11}$$

Since the code rate is a function of SER, the information rate can be defined as $\mathcal{I} = (C_{FSO} + C_{RF}) \cdot \frac{1}{1+P}$. With Eq. (3.9), the total transmission rate of the reliable hybrid system is $C_{FSO} + C_{RF} = C_{FSO} + P \cdot C_{FSO} = C_{FSO}(1+P)$. Thus, the information rate is

$$\mathcal{I} = (C_{FSO} + C_{RF}) \cdot \frac{1}{1+P} = C_{FSO}(1+P)\frac{1}{1+P} = C_{FSO}. \tag{3.12}$$

That means that the system can be optimal and can provide the maximal and constant information rate.

Let us now consider a worst case scenario, whereby the RF channel is affected by a high bit error rate resulting in SER P_{RF} additional to SER P_{FSO} of FSO channel. As a result, $N_{RF}P_{RF}$ and $N_{FSO}P_{FSO}$ out of $N_{FSO} + N_{RF}$ symbols are lost after transmission. For a successful recovery of a sent generation, at least $\tilde{k} = N_{FSO}$ symbols have to reach LNC decoder, i.e., $(N_{RF} - \delta t)P_{RF} + N_{FSO}P_{FSO} = N_{FSO}$. Thus, the number of redundant symbols can be analyzed as

$$N_{RF} = \left\lceil \frac{N_{FSO}(1 - P_{FSO})}{P_{RF}} \right\rceil + \delta t. \tag{3.13}$$

Following Eq. (3.9), the reliable transmission is only possible, if

$$P_{RF} \leq \frac{N_{FSO}(1-P_{FSO})}{N_{RF}-\delta t} \leq \frac{N_{FSO}(1-P_{FSO})}{N_{RF}} \leq \frac{C_{FSO}(1-P_{FSO})}{C_{RF}}, \tag{3.14}$$

The code rate is generally defined by Eq. (3.10) and can be simplified with Eq. (3.14), i.e., $C_{RF} = \frac{C_{FSO}(1-P_{FSO})}{P_{RF}}$, as follows

$$R \approx \frac{C_{FSO}}{C_{FSO}+C_{RF}} = \frac{P_{RF}}{1+P_{RF}-P_{FSO}}. \tag{3.15}$$

Generally, the hybrid system can be optimal and provides the maximal and constant information rate, i.e., $\mathcal{I} = C_{FSO}$. That can be proved as $\mathcal{I} = (C_{FSO}+C_{RF})R = \frac{C_{FSO}(1+P_{RF}-P_{FSO})}{P_{RF}} \cdot R = C_{FSO}$.

Reliable Parallel Transmission

The reliable parallel transmission over N parallel RF/FSO channels allows transmitting the n symbols of an encoded data block in parallel, over N channels, whereby each channel carries exactly one symbol from each generation (Fig 3.2c). Thus, the size of encoded data blocks n equal to the number of parallel channels, $n = N = const$, so that all symbols from the same generation can be sent and received simultaneously. That allows us to increase the effective transmission bit rate and to use the parallel channels at comparably low bit rates, which is more cost-effective. Thus, the number of sources and redundant symbols ($\tilde{k}$ and $\tilde{r}$) needs to be adapted to the channel conditions. Generally, each out of all transmitted symbols from a generation can be corrupted on any FSO channel i, $i \in [1, N-F_{RF}]$, with different SER $p_i \geq 0$, while F_{RF} RF channels are noise-free, i.e. $p_i = 0$, where $i \in [N-F_{RF}+1, N]$. Thus, the overall SER P over all RF/FSO channels is defined as follows $P = \frac{1}{N}\sum_{i=1}^{N} p_i$. Without loss of generality, the parallel channels can be considered as a one cumulative channel with SER P and transmission rate $C = \sum_{i=1}^{N} C_i$. Then, the expected number of erroneous symbols t is determined as

$t = \lceil (n - F_{RF})P \rceil + \delta t$ and the number of required redundant symbols is $\tilde{r} = t = \lceil (n - F_{RF}) \cdot P \rceil + \delta t$. As a result, the code rate of the reliable parallel transmission can be determined as

$$R = \frac{n - \lceil (n - F_{RF}) \cdot P \rceil - \delta t}{n} = \frac{N - \lceil (N - F_{RF}) \cdot P \rceil - \delta t}{N}. \tag{3.16}$$

The maximal code rate in case of FSO channels only, i.e., F_{RF}=0, is

$$R_{max}(F_{RF} = 0) = \lim_{\substack{N \to \infty \\ F_{RF} \to 0}} \frac{N - \lceil (N - F_{RF}) \cdot P \rceil - \delta t}{N} = 1 - P, \tag{3.17}$$

when all N channels are noisy and $P < 1$. When there are $F_{RF} > 0$ noise-free RF channels, the maximal code rate is defined as

$$R_{max}(F_{RF} > 0) = \lim_{\substack{N \to \infty \\ F_{RF} \to N}} \frac{N - \lceil (N - F_{RF}) \cdot P \rceil - \delta t}{N} = 1. \tag{3.18}$$

As a result, the information rate is a function of the total transmission rate C over N RF/FSO channels and the code rate R, $\mathcal{I} = \frac{N - \lceil (N - F_{RF}) \cdot P \rceil - \delta t}{N} \cdot \sum_{i=1}^{N} C_i$.

3.3.3 Performance Evaluation

Next, the analytical results validated by event simulations are presented. The coding process was implemented over a finite field F_{2^8}, i.e., a symbol size was set to $b = 8$ bits. The simulation time was defined as a discrete time, whereby the transmission delay of one symbol over the FSO channel corresponds to one *time unit (tu)*. The transmission rate of FSO channel was set to 10 Gbps, i.e., 1 *tu* = 0.8 *ns* in case of a reliable serial transmission. In the case of N parallel RF/FSO channels, the total transmission rate over all channels was also defined as 10 Gbps, whereby the mean transmission rate C_i of any channel i can be determined as $C_i = \frac{10Gbps}{N}$ and a transmission delay of one symbol as N *tu*. The transmission delay of RF channel in hybrid transmission scenario was defined with Eq. (3.9) as

$\frac{10Gbps}{C_{RF}}$ *tu*. We assumed that all channels have the same propagation delay. The *transmission reliability* was defined as a percentage of successfully decoded generations over all generations sent.

The RF channel was assumed free from noise and atmospheric degradation. In contrast, the FSO channel is affected by the additive white Gaussian noise, which followed Bernoulli distribution, whereby the bit error rate (BER) and, consequently, the symbol error rate (SER) were varied as $[1.25 \cdot 10^{-4}, 1.3 \cdot 10^{-2}]$ and $[10^{-3}, 10^{-1}]$, respectively. The conversion from BER into SER was defined as $SER = 1 - (1 - BER)^b$. Generally, the symbol loss on the FSO channel could be balanced out with additional redundant symbols. Thus, each corrupted dropped symbol was replaced by the redundant symbols from the same generation. Thereby, the transmission reliability of any proposed transmission method could reach up to 100% without a need for retransmission, when the FSO channel has a nearly constant SER (up to $SER \leq 10^{-1}$) resulting in single symbol errors only. The 99% confidence intervals present the values of the range of 10^{-4} and were not shown for clarity.

Now, a scenario where the quality of the FSO channel dramatically decreases and then fluctuates is investigated. Generally, as long as the SER is constant, the required number of redundant symbols for reliable transmission is known and set to $r \geq t = \tilde{t}$ and $\delta t \geq 0$, while the symbol error variation δt was set to 0 at the beginning. The results illustrate our observation of the FSO channel during the first 400 *tu*, while SER was changed every 10 *tu*. The SERG analyzed the incoming data blocks, counted the number of erroneous symbols and calculated the variation of symbol errors with Eq. (3.2). To study the minimal code rate, we did not define any thresholds for triggering a transmission stop in case of a very high SER.

Fig. 3.3 shows the change in the code rate of the reliable serial and parallel transmission with increasing SER. In the case of serial transmission, the sender uses a constant number of source symbols

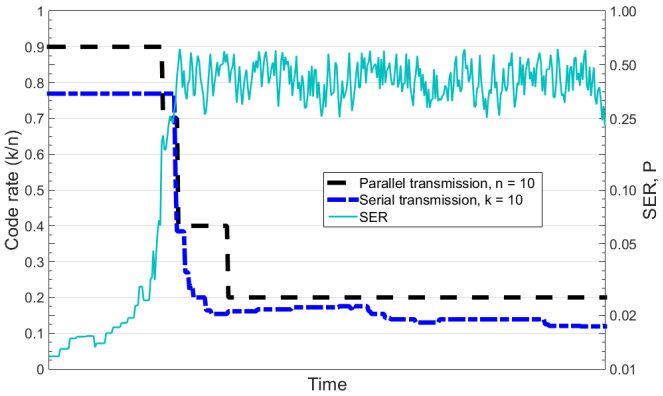


Figure 3.3: Code rate of reliable serial and parallel transmission [4].

$k = 10$ and changes the number of coding redundancy as $\tilde{r} = \tilde{t} + \delta t$ according to Eq. (3.4). The reliable parallel transmission system uses $N = n = 10$ parallel FSO channels, with a total transmission rate of 10 Gbps. As a result, the sender transmits $n = 10$ symbols from the same generation in parallel every 10 *tu*. In this case, the number of source symbols is not constant and depends on the number of predicted redundant symbols $\tilde{r} = t$. In both scenarios, the maximal code rate of 78% and 90% is reached for SER$\leq$ 5%, respectively. The code rate dramatically decreases with increasing SER and reaches the minimal values of 12% and 20% for SER between 25% and 62%, respectively. In contrast to the reliable serial transmission, the code rate of the reliable parallel transmission rapidly reached a steady state in case of fluctuating value of SER.

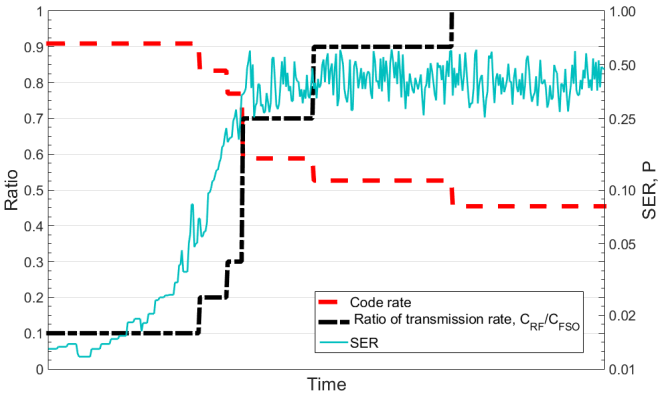


Figure 3.4: Code rate of reliable Hybrid RF/FSO transmission [4].

Fig. 3.4 presents the changes of SER, the code rate and required transmission rate of RF channel in case of a reliable hybrid transmission over $N = 2$ channels, i.e., FSO and RF. Here, the number of source symbols $\tilde{k}$ as well as the number of redundant symbols $\tilde{r}$ are predicted by SERG according to Eq. (3.8) and Eq. (3.7) and sent over FSO and RF channels, respectively. Moreover, we assume that the RF channel is able to change its transmission rate, which can be defined according to Eq. (3.9). The main goal, however, is to reach the maximal code rate and the maximal information rate. Generally, the FSO channel demonstrates the best performance when the information rate keeps constant as $\mathcal{I} = C_{FSO} = 10$ Gbps according to Eq. (3.12). As a result, the code rate is higher than 46%, which is a minimum in case of high and fluctuating SER. On the other hand, the required transmission rate of the RF channel can become larger than one of the FSO channels, which is impractical.

3.3.4 List of Symbols

P	symbol error rate;
R	code rate;
$\mathcal{I}$	information rate;
C	transmission rate;
N	number of RF or FSO interfaces, i.e., channels;
F_{RF}	number of RF interfaces, i.e., channels, out of N;
k	number of uncoded source symbols;
r	number of redundant coded symbols;
n	size of coded data blocks, i.e., $n = k + r$;
t	expected number of erroneous symbols per n symbols;
b	symbol size in bits.

3.4 Reliable Network Function Chaining as a Service

In this section, we consider NFV services as an ordered sequence of network functions, i.e., VNFs, so-called SFC. Each VNF has a particular place in the order of functions building SFC, whereby the position of a certain VNF in SFC defines the VNF type. In contrast to this assumption, SFCs can be built by VNFs with different or the same functionality. To complete any requested service, all VNFs of the composed SFC have to be available during service run-time and each of these VNFs needs to complete the processing successfully. Thus, in this section, we refer to successful service completion as a *service success* and to a probability of service success as *SFC reliability*. Without loss of generality, we refer to VNFs, which read or write packet header only, as *header-VNFs* (h-VNF) and to VNFs, which read or write packet payload, as *payload-VNFs* (p-VNF). Most of the existing VNFs are h-VNFs. Tab. 3.2 shows some typical network functions and their processing on packets. The packets can be registered by passive probing function (Probe), the Firewall (FW) proves if traffic is allowed to pass through DCN. The legitimate packets from FW are processed by Network Address Translation (NAT), which replaces the IP addresses and port numbers in the packet header. Packets can be processed by the Intrusion Detection System (IDS), which copies the packet flow to perform offline traffic analysis to identify and log violations. The Traffic Shaper (TSp) categorizes and queues the packets to meet a certain quality of service (QoS). Thus, only NAT changes the header and only IDS is p-VNF, i.e., looks at the payload. These VNF functionalities need to be considered by building SFCs and by selecting a strategy for VNF protection against failures. Next, we introduce the concept of a reliable network function chaining and analyze the service success.

VNF:	Probe	FW	NAT	IDS	TSp
Packet Header	R	R	R/W	R	R
Packet Payload	-	-	-	R	-

Table 3.2: Different virtual network functions and the related processing of IP packets, i.e., R: read; W: write, [6].

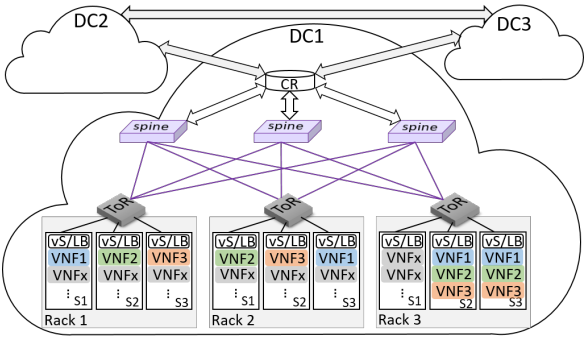


Figure 3.5: DCN based on leaf-spine architecture [5].

3.4.1 Service Function Chaining in DCN

The VNFs of any SFC are typically hosted by Virtual Machines (VM) installed on servers in data centers (DC), whereby several servers are collected in the same rack and any DC provides multiple racks [83, 84, 85]. The DCN architecture is generalized as a hardware fabric of switches, links, racks with multiple servers and servers with multiple VMs as illustrated in Fig. 3.5, where DC1 has the leaf-spine DC topology. To complete the requested service, traffic needs to pass all VNFs of a certain SFC. That can be implemented with SFC Encapsulation as proposed in [86], whereby each SFC is characterized by Service Function Path (SFP) with a specific ID, i.e., additional Network Service Header with Service Path Identifier [87]. In Fig. 3.5, the ordered sequence of VNFs, i.e., SFC (VNF1-VNF2-VNF3), can be built in the same server, e.g., in S2 or

S3 in Rack 3, or distributed over different servers, e.g., over S1-S3 in Rack 1 or over S1-S3 in Rack2. There are multiple options to build SFCs over different racks as well, e.g., VNF1 can be utilized in S1(Rack1), VNF2 and VNF3 in servers S1(Rack2) and S2(Rack3), respectively. A placement of VNFs from the same SFC in different DCs is possible as well. To provide the SFP, there is a need for connectivity between all relevant DCN components, i.e., racks, servers, VNFs, and between DCs. Thus, we assume that any DCN topology includes four types of switches: core router (CR), Top-of-Rack (ToR), forwarding switches, e.g., spine, and Virtual Switch (vS). Core router provides connectivity between Wide Area Network (WAN) and DCNs and, thus, routing services to other parts of own data center and connectivity to different remote locations outside. Since each rack incorporates its own subnet of multiple servers, ToR switches handle forwarding within a rack. The forwarding switches provide a connection to the core router and forward traffic between racks based on segment routing (SR), whereby ToR switches are fully meshed to a series of forwarding switches to ensure low latency and low likelihood of congestion inside DCN. The traffic towards other racks is routed by the source ToR, which addresses the destination ToR by packet labeling. The forwarding switches perform only label lookups to route the traffic. SR allows changing the end-to-end path utilized over the network, whereby the source ToR switch needs only to change the label. That is an important capability to provide maximum flexibility in case of component failures and the need for traffic redirection. Each server can host multiple VMs, whereby each VM reserves as many resources as required for one VNF to serve one user request. All VMs are connected to vS, e.g., programmable hypervisor switch with a load balancer (LB), which selects one VNF for processing of arrived packets. Thus, the main task of vS is the selection of a legitimate VNF instance for any arrived request to provide a connection to the ToR switch.

3.4.2 Parallelism in DCNs

We refer to a large amount of IP packets with the same IP addresses and port numbers in the header, which pass the same SFC, as *traffic flow*. Large traffic flow can be split into parallel sub-flows and uniformly distributed over the network. Traffic can be spread over parallel SFCs using Equal Cost Multipath Protocol (ECMP), a well-known technique to improve load balancing in servers and the network, i.e., to avoid server and links overload. We model parallelism in three distinctive ways: 1) Traffic Parallelism: the serial traffic flow is split into k parallel *sub-flows*; 2) SFC Parallelism: all VNFs of the requested SFC are replicated into at least k VNF instances to process k sub-flows, resulting in *parallelized* SFC presented by k parallel the so-called *sub-SFCs*; and 3) Path Parallelism: all k sub-flows are independently transmitted over k link disjoint paths and processed in parallel by k parallel sub-SFCs.

As shown in Fig. 3.6, any traffic flow f can be split (parallelized) into $k = 3$ sub-flows, i.e., f_1, f_2 and f_3. The SFC of $\Psi = 3$ VNFs is then parallelized into $k = 3$ sub-SFC. The sub-SFCs are placed in the same DC following different placement strategies. Sub-SFC for f_1 is distributed over several servers inside Rack3 (S4, S5 and S7). All VNFs of sub-SFC for f_2 are placed in the same server S1 in Rack1. VNFs of sub-SFC for f_3 are distributed over different racks. As a result, each sub-flow can utilize its own path through DCN. The end-to-end service can be successfully completed, when all three sub-flows can be processed by VNF1, VNF2 and VNF3, i.e., all parallel sub-SFCs are available during service run-time. Next, we discuss in detail the three concepts that we apply to DCN parallelism.

Traffic Parallelism

Traffic spreading is a well-known technique to improve load balancing on the network interfaces [17, 18]. To realize traffic parallelism

via traffic spreading, however, the incoming traffic needs to be sorted into different flows based on source and destination IP addresses and port numbers in packet headers. That is because different traffic flows would require different SFCs. Finally, packets of any flow f can be distributed over k interfaces in Round Robin fashion resulting in k parallel sub-flows. When outgoing DCN traffic needs to be sent serially, after service completion by the certain SFCs, all packets from all flows and sub-flows need to be serialized into one data stream based, for instance, on FIFO.

SFC Parallelism

Generally, parallel sub-flows can be sent to either the same VNF instances of an SFC or separate replicas of an SFC, consisting of certain VNFs, created for each sub-flow, which we call SFC parallelism. For the latter, all VNFs of a certain SFC are replicated into k VNF instances resulting in k parallel sub-SFCs. As a result, each sub-flow passes its sub-SFC as shown in Fig. 3.6. That can be implemented with SFC Encapsulation as proposed in [86], whereby each parallel sub-SFC can be characterized by Service Function Path with a specific ID. For that reason, each packet requires an additional Network Service Header with Service Path Identifier of 24 bits as discussed in [87]. It is necessary that all k VNF replicas of a certain type process packets of a certain traffic demand based on the same rules and with the same processing result. For instance, if one of the VNF types represents a NAT, all k NAT replicas have to replace IP addresses and ports in the packet headers independent of sub-flow by the same predefined IP addresses and ports. Thus, the parallel VNFs need a management unit for its coordination and rule update. When flow and SFC parallelism are deployed, the synchronization between VNFs of the same type can be provided by an external state repository, which can store internal states of VNFs [26].

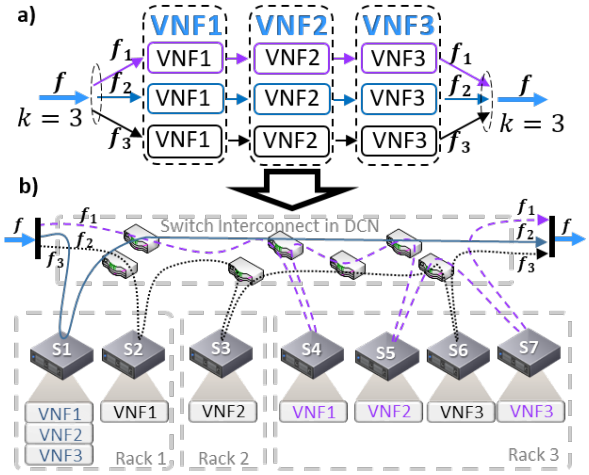


Figure 3.6: Deployment of parallelism [6]: a) Parallelism of Traffic and SFCs; b) Parallelism in DCN and VNF placement.

Path Parallelism

In general, to reach better reliability, VNFs from different sub-SFCs should not share the same DCN components. That prevents the usage of the same links and path segments by parallel sub-flows and, thus, additionally increases reliability. Then, after traffic parallelization, the k parallel sub-flows should be sent over k parallel link and server disjoint paths, i.e., through k maximally disjoint sub-SFCs. Then, each parallel sub-flow passes through its sub-SFC. Fig. 3.6b) illustrates a possible sub-SFCs placement within DCN, whereby VNFs of any sub-SFC for a certain sub-flow are placed disjoint from VNFs of other sub-SFCs, e.g., sub-SFC of sub-flow f_3 is placed in servers S2, S3 and S6 and disjoint from other sub-SFCs placed in S1, S4, S5 and S7. Any flow is successfully processed by a certain SFC when all parallel sub-flows passed its sub-SFC and meet in the same server to be serialized to leave DCN. Without loss of generality, any path can be considered as a composition of path

segments that connect different servers. As shown in Fig. 3.6b), each sub-SFC can be placed in up to three different servers. Thus, the path for sub-flows f_1 and f_3 consists of four path segments, any of which can fail. Failure of path segments between S2 and S3 is equivalent to a failure of server S3 and VNF2 of the third sub-SFC.

3.4.3 Component Failures

Without loss of generality, VNFs, servers, racks and DCs are the atomic components required to build an SFC. Any component failure can result in complex failure propagation as described next:

- Data Center. The failure of the data center can be caused by a failure of connectivity through CR, failure of a core router or connection between the core router and all forwarding switches. Failure of DC will result in the failure of all racks, servers and VMs relevant for building and maintenance of SFC.

- Rack. Rack failures can be a result of a failure of forwarding or ToR switches as well as a failure of links between them. Any failure of a rack will cause the failure of all servers within it and all VNFs hosted by these servers.

- Server. Any server can be considered as failed (unavailable) when hardware components, i.e., power supply, memory, etc., of the server or link between server and ToR or vS/LB are failed, whereby all VMs installed on the server will fail as well.

- VM/VNF. Since we assume that each VM reserves as many resources as required for one VNF to serve the incoming requests, the failure of any VM causes a failure of one VNF only. We assume that VMs on the same server are separated so that their failure has not any impact on any other VMs, i.e., VNFs.

- Path Segment. When any path is considered as a combination of path segments, which connect different servers with relevant

VNFs, we assume that the DC and racks are always available. The failure of any path segment results in no connectivity to a certain server and is equivalent to the server failure and failure of all VNFs running on this server.

3.4.4 Failure Protection Strategies

Without any additional resources, any parallelized SFC is characterized by a certain level of reliability, which depends on the reliability of the underlying DCN components, i.e., DCs, Racks, Servers and VMs and path segments. To increase the probability of service success, i.e., SFC reliability, two failure protection methods are studied next: backup protection and protection with erasure coding.

Failure Protection with Backup

To additionally protect the active VNFs over all parallel active sub-SFCs, any SFC can be enhanced by r backup VNFs resulting in $n = k + r$ parallel sub-SFCs. The backup VNFs can then replace any failed active VNF of the same type over any sub-flow. Similar to active VNFs, also backup VNFs can fail during service run-time.

As shown in [26, 88, 75], to provide highly reliable communication, there are multiple different strategies for active monitoring of NFV, efficient hardware and software failure detection and effective deployment of active and backup VNFs. The most effective VNF deployment is to utilize multiple switches, e.g., vS, ToR and forwarding switch that can detect a failure in hard- and software and redirect traffic to the corresponding available backup VNF if an active VNF fails. However, here, a backup deployment strategy is not explicitly considered. Instead, the assumption is that any service is successful if at least $k \geq 1$ out of n VNFs of each type are available during service run-time while at most r active or backup VNFs of any type can fail without causing service interruption. Depending on

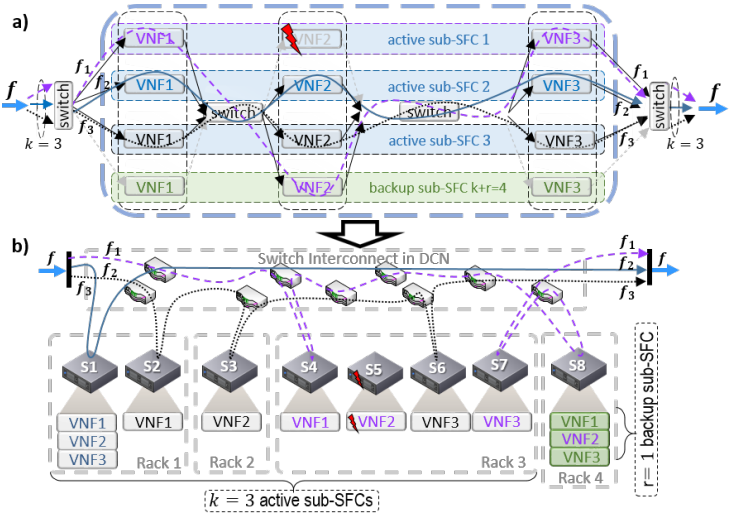


Figure 3.7: Backup protection and placement of sub-SFCs [6]: a) VNF deployment. b) Network view.

the SFC placement strategy, a failure of different component types will lead to a different number of failed VNFs. It should be noted that the synchronization between active and backup VNFs of the same type and, thus, an external state repository is required. Any backup VNF would then start its operation from the "reset" state. Then, any critical state of the failed original VNF can be stored and retrieved from the external state repository, and that up to the point of its failure as discussed in [26].

Fig. 3.7 illustrates the idea of a possible deployment and placement of backup sub-SFC in DCN. The active SFC is parallelized into $k = 3$ active and $r = 1$ backup sub-SFCs. Just like an active sub-SFCs, a backup sub-SFC is placed in a server S8 in Rack4 providing backup VNFs. The $k = 3$ parallel sub-flows are sent over 3 parallel disjoint paths through 3 sub-SFCs. Since VNF2 of sub-SFC1 fails, the switch inside the DCN redirects sub-flow f_1 to VNF2 of backup

sub-SFC 4. Thus, the sub-flow f_1 is redirected to backup server S8, where VNF2 is activated to replace the original VNF2 from server S5. As a result, f_1 is processed by backup VNF2 in Rack 4 and then send to active VNF3 on S7 to complete the service.

Failure Protection with Coding

The use of *erasure coding* such as random LNC is a classical solution to improve the reliability of transmissions. Erasure codes can add redundancy into data to protect it against losses. Any (n, k) erasure code encodes k units of original data into n units of coded data, in which any k out of n units can recover the original data and, thus, the code can tolerate the failures of up to $r = n - k$ data units. We refer to any k units of original data encoded together as *generation* and to r additional data units generated by encoding as *redundancy*. The erasure codes can be deployed in practice as *systematic codes*, meaning that the k original data units are unchanged after encoding and only the r redundant data units present a linear combination of k original data units. Thus, that k data units of coded data can be directly accessed without prior decoding. In contrast, when at least one data unit out of k unchanged data units is lost, any out of r redundant units from the same generation replaces the lost data unit, while decoding is required to recover k original data units. The decoding is only successful if at least k data units from each generation arrive at the decoder. For VNF protection with coding, we assume that each parallel sub-flow can be presented as a sequence of data units, whereby the original and encoded data units have the same predefined size. We refer to the k parallel sub-flows, which are unchanged after encoding, as *main sub-flows*, and to the sub-flows, which are built by redundancy as *redundant sub-flows*.

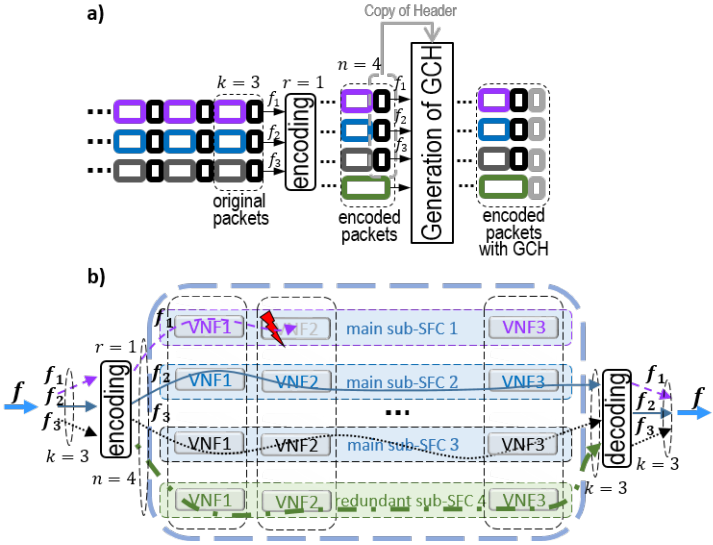


Figure 3.8: Protection with systematic erasure coding [6]: a) packet encoding process and generation of GCH; b) VNF deployment in case of VNF failures.

To be able to recover both header and payload, both need to be encoded, i.e., the whole packets, which are interpreted as a data unit by an encoder. An example of the encoding process is illustrated in Fig. 3.8a), where $k = 3$ parallel sub-flows encoded to generate $r = 1$ redundant sub-flow. The three original packets, one from each parallel sub-flow, build generation and are combined into the green redundant packet. In contrast to original packets, which are unchanged after encoding, the redundant packet does not have any header or readable payload. Redundant packets belong to the same flow as original packets and need to pass through the same chain of VNFs. Usually, VNFs are only able to process uncoded data. Thus, there is a need for a new header for generated redundant packets in accordance with the original header. Thus, we propose to utilize

a *generalized coding header* (GCH). As presented in Fig. 3.8a), the original headers are copied to generate a certain GCH with the same source/destination addresses and the same port numbers as determined for the original traffic flow. The generator of GCH (Generation of GCH) utilize a copy of original IP and TCP headers and change some header fields: 1) In the IP-GCH, the Header Length, Total Length and Header Checksum are adapted to the coded packet length; 2) In the TCP-GCH, data offset and header checksum needs to be changed. Moreover, the sequence numbers need to be redefined to identify the packets from the same generation. Finally, the GCHs (gray) are attached to the certain encoded packet. Additionally, the metadata can be inserted into each packet to hold information related to the coding process, e.g., the number of main and redundant sub-flows, generation size, etc. The information from GCH is copied to the original header after any flow recovery or any decoding. Without loss of generality, the encoding (decoding) process requires packet buffering and clocking to build generations. That can be implemented by VNFs such as Traffic Shaper.

Fig. 3.8b) shows the VNF deployment for the failure protection with a systematic erasure code. The original flow f is parallelized into $k = 3$ sub-flows. The parallel sub-flows are encoded into $n = 4$ sub-flows providing $r = 1$ redundancy. That requires $r = 1$ redundant sub-SFCs. Thus, there are $k = 3$ main sub-flows after encoding, f_1, f_2 and f_3, while redundant sub-flow presents a linear combination of them. After encoding, the sub-flows are sent over $n = 4$ disjoint paths through $n = 4$ parallel sub-SFCs. The VNF2 of the main sub-SFC1 fails, resulting in loss of the main sub-flow f_1. In contrast to backup protection, there is no need for neither VNF migration nor traffic redirection. The blue (f_2), black (f_3) and green redundant sub-flows, i.e., $k = 3$, arrive at destination. A recovery of original flow f is possible by decoding. Generally, the decoding is only required, when at least one out of k main sub-flows is lost.

Required Challenges in Implementation: Since different network functions read/write various parts of the packet during its operation, header, payload, or both, the main challenge of coding protection is to provide coded packets that can be processed by both h-VNFs and p-VNFs. Generally, any h-VNFs can process (read/write) GCH and, thus, coded redundant packets. In contrast, p-VNFs such as IDS can not work with coded packets and require decoding prior to packet processing. As a result, p-VNFs are not able to process coded packets and can be protected by backup only.

Hybrid Backup-Coding Failure Protection

Since any SFC generally consists of both h-VNFs and p-VNFs, which process packet header and packet payload, respectively, a hybrid backup-coding protection of SFC is introduced next. An example of hybrid VNF protection is presented in Fig. 3.9, where VNF4 is only a p-VNF, which needs prior decoding and backup protection. After traffic parallelization into $k = 3$ original sub-flows, i.e., f_1, f_2 and f_3, and encoding them into $n = 4$ coded sub-flows. All $n = 4$ sub-flows are sent over disjoint paths through different sub-SFCs. The purple sub-flow (f_1) is lost due to the failure of connection to S3 or failure of S3 or VNF2 failure. The 3 remaining sub-flows (blue, black and green) are successfully processed by the first three h-VNFs of related sub-SFCs in servers S1, S4 and S11. Since the main purple sub-flow is lost, the decoding is required in S8 to recover all 3 original sub-flows (purple, blue and black) before processing by p-VNF4 representing IDS. During decoding, the green redundant sub-flow replaces the lost purple sub-flow. The recovered original sub-flows are sent through different instances of VNF4 placed in servers S8, S9, S10. The $r = 1$ backup VNF4 in server S5 can replace any failed active VNF4 requiring sub-flow redirection. Finally, the original sub-flows are encoded in S9 into $n = 4$ sub-flows and sent to h-VNFs of type 5. The black VNF5 in server S7 fails and causes

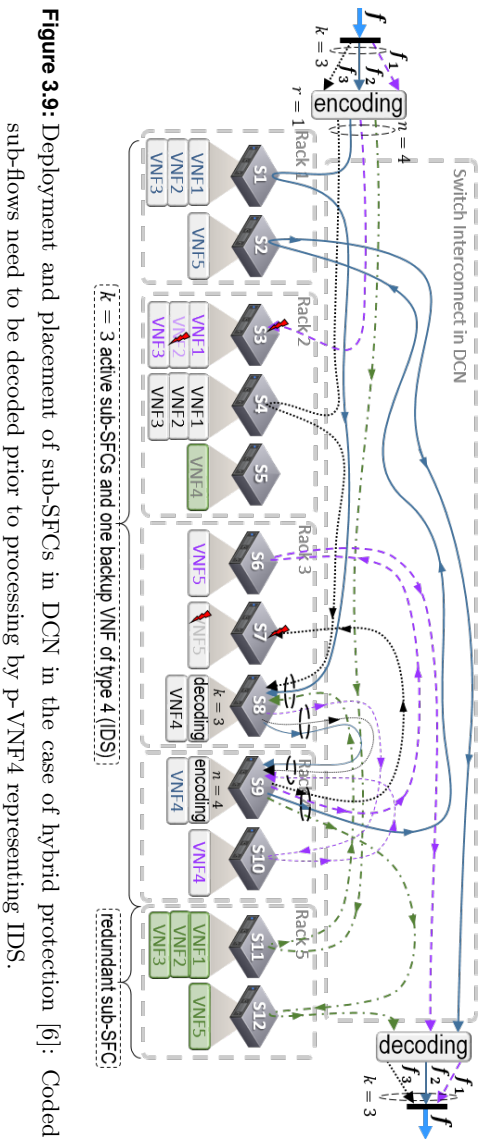


Figure 3.9: Deployment and placement of sub-SFCs in DCN in the case of hybrid protection [6]: Coded sub-flows need to be decoded prior to processing by p-VNF4 representing IDS.

the loss of the main black sub-flow (f_3) and in a need for decoding at destination prior to traffic serialization. The decoding is possible as f_1, f_2 and redundant sub-flows reach decoder.

3.4.5 Modeling of parallelized SFC

For assessing any SFC placement strategy as well as the placement and component dependent SFC reliability, we introduce next some concepts for modeling and tracking the interdependency of component failures. Used notations are shown in Sec. 3.4.8.

Modeling of Component Interdependency

Embedding VNFs of any SFC within a DCN is a challenging issue of building virtual networks with different objectives such as improving network survivability, SFC reliability, load balancing, network utilization or reduction of resource cost. As illustrated in Fig. 3.10a), an SFC is replicated into two sub-SFCs and a traffic demand for the parallelized SFC is split into $k = 2$ sub-flows presented by solid and dashed lines. Each sub-SFC can be allocated to DCN components based on different placement strategies. In the example, one sub-SFC is placed in a DC4, and VNFs of another sub-SFC are distributed over DC1, DC2 and DC3 connected through WAN. Moreover, there are multiple options to place VNFs inside DC. For instance, in DC4, sub-SFCs are placed in the same or different racks and servers. Also, VNFs of a certain sub-SFC can be distributed over several servers or racks. Here, Rack1 in DC4 accommodates a whole sub-SFC, which is distributed over servers S1, S2 and S3, while, in Rack2, two SFCs are distributed over three servers, i.e., each server provides two VNFs of the same type. In contrast, in Rack3 (DC4), one whole sub-SFC is allocated to one server. In contrast, each VNF of a sub-SFC distributed over DC1, DC2 and DC3 is placed in separated racks and servers.

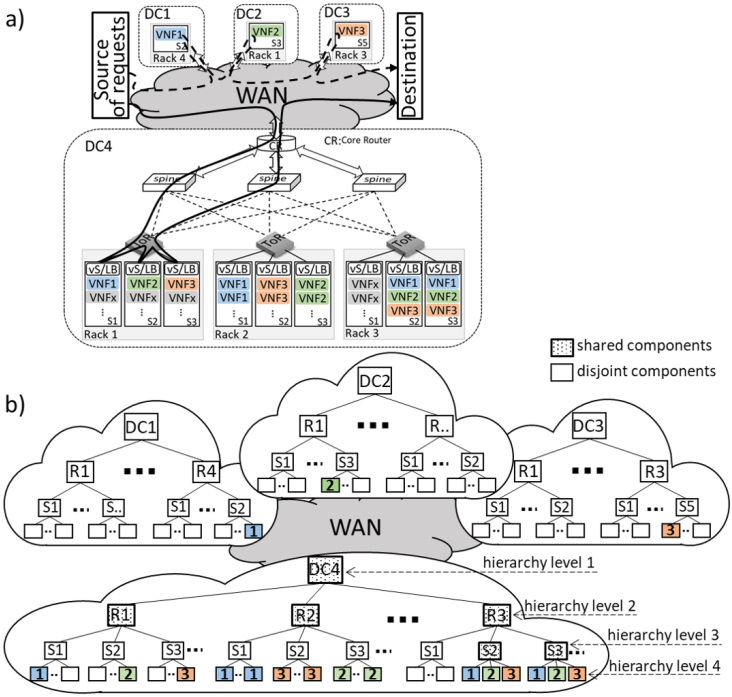


Figure 3.10: Placement of SFCs over Inter- and Intra-DCN [7]: a) SFC placement and deployment in DCN; and b) Hierarchical placement and component interdependency.

As the reliability analysis requires tracking of the interdependency between component failures, the DCN is modeled as a hierarchical tree, as shown in Fig. 3.10b), where failures of DC, rack, server and VNF are possible. The root node is a DC, which belongs to the highest hierarchy level. The VNFs represent leaf nodes and belong to the lowest hierarchy level. The links between any nodes define interdependency between components. Without loss of generality, there are $\mathcal{C}$ component types in the hierarchical tree with $\mathcal{C}$ hierarchy levels. Each hierarchy level is described by a variable c, $1 \leq c \leq \mathcal{C}$, and determines the component type. When the SFC of length Ψ

is distributed over $\mathcal{C} = 4$ component types, any components of type $c = 4$ and type $c = 1$ belong to the lowest and the highest level in the hierarchy, respectively. Considering different placement strategies in Fig. 3.10, it can be seen that some VNFs are allocated to the same components, e.g., DC4 and rack 2 (R2), and, thus, share these components. We refer to any component which is utilized by all VNFs from the same SFC, i.e., by VNFs of different types, as **shared component**. In Fig. 3.10b), all shared components are shown by dotted blocks. The SFCs in DC4 have a different number of shared components, i.e., SFC placed in R1 has two shared components (R1 and DC4), similarly VNFs of SFCs placed in R2 share R2 and DC4. The SFCs in S2 and S3 (R3) have three shared components each, i.e., server (S2 or S3), R3 and DC4.

The opposite of shared components is the *disjoint component* that separate VNFs of a certain (sub-)SFC. Thus, we introduce the **level of disjointness** Δ, where $1 \leq \Delta \leq \mathcal{C}$, which describes the number of hierarchy levels unshared by VNFs from the same SFC, i.e., a number of component types provided to place a certain VNF type disjointly from other VNF types. Since any VNF is placed in a separate virtual machine, i.e., disjointly from any other VNFs, the minimal level of disjointness is defined as $\Delta = 1$. For instance, $\Delta = 1$, when different VNF types are placed in the same server, where VNFs are disjoint only, such as illustrated in Fig. 3.10, i.e., SFCs placed in R3 (DC4). The level of disjointness of SFC placed in DC1, DC2 and DC3 is $\Delta = \mathcal{C}$, where different VNF types are placed in different DCs, racks, servers and VMs and have no shared components. Generally, Δ means that Δ component types from lower hierarchy levels, i.e., from level $\mathcal{C} - \Delta + 1$ to $\mathcal{C}$, are pre-reserved for a certain VNF type, e.g., VNF1, and are disjoint from all other components of the same type c, which are pre-reserved for another VNF type, e.g., VNF2. We can also derive a type of shared components, i.e., their hierarchy level, which varies from $\mathcal{C} - \Delta$ to 1, whereby the

number of shared components is $(\mathcal{C} - \Delta)$.

To be able to characterize any SFC placement more precisely, we next introduce a **heterogeneity degree** N_r, where $1 \leq N_r \leq \mathcal{C}$. The heterogeneity degree shows how many different component types are utilized to separate VNFs of the same type, e.g., any VNF3 from all other VNFs of type 3 from the same parallelized SFC. When VNFs of a certain type are placed in different data centers, racks, servers and VMs, the heterogeneity degree is $N_r = 4$. In contrast, placement of VNFs of a certain type in the same server, i.e., in different VMs only, reduces the heterogeneity degree to $N_r = 1$. In Fig. 3.10, the SFC placement in R3 (DC4) results in $N_r = 2$ as each VNF of a certain type, e.g., VNF2, has its own VM and server, i.e., both VNFs2 are placed disjointly in S2 and S3. However, the SFC placement in R2 (DC4) results in $N_r = 1$ as VNFs of the same type are allocated to the same server.

In the presented tree architecture, the components of the same type, i.e., of the same hierarchy level, are independent of each other. The components of the lower hierarchy level do not affect connected components of the higher hierarchy level. For example, a failure of VNF1 of SFC placed in R1 (DC4) will affect the availability of neither VNF2 and VNF3 nor servers S2 and S3. The availability of S1, Rack1 or DC4 will not be affected by VNF1 failure as well. In contrast, the components of higher hierarchy level impact connected components of lower hierarchy level, e.g., a failure of R1 (level 2) will result in the failure of all servers (level 3) inside this rack and all VMs (level 4), i.e., failure of all VNFs (VNF1, VNF2 and VNF3). Additionally, also failures of whole SFCs due to some component failures need to be considered. As shown in Fig. 3.10b), all VNFs from any SFC in DC4 have some shared components, i.e., SFC placed in R1 has two shared components (R1 and DC4), VNFs of SFCs placed in R2 share R2 and DC4 and the SFCs placed in S2 and S3 (R3) have shared server (S2 or S3), R3 and DC4. As a result, a

failure of any shared component will result not only in the failure of other components from a lower hierarchy level but also in the failure of an entire sub-SFC, i.e., Ψ VNFs of different types.

SFC and VNF Placement

In our model, any VNF of a certain sub-SFC is placed in a similar fashion requiring the same number of DCs, racks and servers. To reduce the number of backup resources and to provide a level of disjointness, different strategies are possible. For latency reduction, VNFs of different types can be placed together to build an SFC, i.e., VNF placement based on SFC. If VNFs of the same type are placed together, i.e., VNF placement based on VNF type, the control and management of Service Function Paths are simplified [86]. Using hierarchical tree concept with $\mathcal{C} = 4$ hierarchy levels, to place all $n = k + r$ VNFs of a certain type, where $k \geq 1$ and $r \geq 0$, there is a need for n_1, $1 \leq n_1 \leq n$, DCs, n_2, $1 \leq n_2 \leq n$, racks inside any DC, i.e., in total $n_1 n_2$ racks, and $1 \leq n_3 \leq n$ servers inside any rack, i.e., in total $n_1 n_2 n_3$ servers, and $k + r$ VMs, whereby any server can host n_4, $1 \leq n_4 \leq k + r$ VNFs of the same type. Thus, we introduce a configuration set $\epsilon = \{n_1, n_2, n_3, n_4\}$ to determine a certain placement strategy utilized by several sub-SFCs and the required number of components of any type. For the example in Fig. 3.10, the configuration sets can be defined as $\epsilon = \{1, 1, 1, n\}$, as $\epsilon = \{1, 1, n, 1\}$ or as $\epsilon = \{1, 1, 1, 1\}$ for SFCs placed in DC4, i.e., in R2, in R3 or in R1, respectively.

Fig. 3.11 illustrate all 16 possible placement strategies (Pl.) to symmetrically distribute n VNFs of any type, i.e., from n sub-SFCs, over Inter- and Intra-DCNs. For example, in Pl. 1, each VNF is placed in a different DC, where each DC requires $n_2 = 1$ rack and $n_3 = 1$ server to accommodate $n_4 = 1$ VNF resulting in configuration $\epsilon = \{n, 1, 1, 1\}$. Since VNFs of the same type are separated by all component types, i.e., DC, R, S and VNFs, the heterogeneity degree

Figure 3.11: Inter- and Intra-DCN placement of $n = 2$ sub-SFCs utilizing following components: data center (DC), rack (R), server (S) and VNF (presented by numbers) [5].

is $N_r = 4$. Besides, the level of disjointness is $\Delta = 4$ since there are no shared components. In contrast, Pl. 2 allocates all VNFs to the same DC, which is a shared component, i.e., $\epsilon = \{1, n, 1, 1\}$, resulting in $\Delta = 3$. The heterogeneity level is also reduced to $N_r = 3$ as VNFs of the same type are placed in the same DC. Based on the basic configurations from Pl.1 to Pl.4 we can distinguish two strategies. In the case of " VNF placement based on SFC", whereby VNFs of different types are placed together as shown by the placements Pl. 5, Pl. 7, Pl. 8, Pl. 11, Pl. 12 and Pl. 13. On the contrary, the placements Pl. 6, Pl. 9, Pl. 10, Pl. 14, Pl. 15 and Pl. 16 are based on the "placement based on VNF type", whereby VNFs of the same type are placed together. The presented example can be extended to any value of n and any number of VNF types inside SFC. In contrast, in practice, the placement of active and backup sub-SFCs is a combination of different placement strategies shown in Fig. 3.11 and requires further classification and modeling as presented next.

SFC Reliability Classes

Without loss of generality, each sub-SFC build a reliability class ξ. There can be at most $N = n = k + r$ reliability classes, i.e., $1 \leq \xi \leq N$. As a result, the component dependent reliability of each component type c can be denoted as p_{c_ξ}, where ξ shows the reliability class of any component type c utilized to accommodate VNFs from this reliability class. Fig. 3.12 illustrates an example of parallelized SFC of three VNFs (VNF1-VNF2-VNF3) replicated into $k = 3$ active sub-SFCs and one backup sub-SFC (VNF1b-VNF2b-VNF3b). The active sub-SFCs are placed in DC7, two of them in Rack1. The third active sub-SFC is distributed over three different racks in DC7 so that each VNF has its own rack, server and virtual machine. Each backup VNF of the backup sub-SFC is placed in individual DC, i.e., DC1, DC2 and DC3. In this example, all $n = 4$ sub-SFCs belong to different reliability classes ξ, i.e., $N = 4$, whereby each sub-SFC

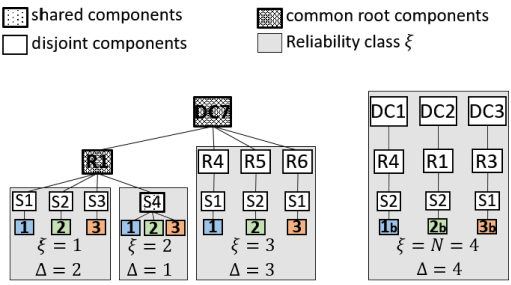


Figure 3.12: Hierarchical structure of complex service function chaining for $k = 3$ and $r = 1$ and reliability classification based on placement strategy [7].

will have certain reliability different from the reliability of any other sub-SFC also when all component dependent reliabilities are equal. That is due to different placement strategies utilized, i.e., a different total number of components involved to place each sub-SFC.

To simplify the reliability analysis, we assume that all components of type c utilized to host Ψ VNFs of a certain sub-SFC have an equal component dependent reliability p_c, i.e., $p_c^1 = ... = p_c^\psi = ... = p_c^\Psi = p_c$, where $1 \leq c \leq \mathcal{C}$ and ψ, $1 \leq \psi \leq \Psi$, shows the VNF type placed in component c. Thereby, the component dependent reliability of components from different hierarchy level does not need to be the same, i.e., $p_{c-1} \neq p_c \neq p_{c+1}$. Thus, all DCs, all racks, all servers and all VMs utilized by, e.g., backup sub-SFC in Fig. 3.12, have equal values of component reliability, respectively. In contrast, each sub-SFC can have components with reliability different from component reliability utilized by any other sub-SFCs, i.e., $p_{c_\xi} \neq p_{c_{\xi+1}}$.

For a successful service completion, any k out of n available VNFs of each type from any sub-SFCs are required to be available during service run time, i.e., there is no need to differentiate between reliability classes for service success. Generally, several sub-SFCs may be combined into a certain reliability class, which further simplifies reliability analysis. Multiple sub-SFCs can belong to the same reliability class only when any components involved are character-

ized by the same hierarchical interdependency determined through placement strategy and component dependent reliability. As a result, if there are n', $1 \leq n' \leq n$, sub-SFCs, which are hosted by components with the same component reliability, i.e., $p_{c_\xi} = pc_\xi + 1$, $p_{c+1_\xi} = pc + 1_\xi + 1$, etc., and all n' sub-SFCs are placed following the same placement strategy, resulting in the same level of disjointness $\Delta_\xi = \Delta_{\xi+1}$ and heterogeneity degree $N_{r_\xi} = N_{r_{\xi+1}}$, then all $n' \equiv n_\xi$ SFCs belong to the same reliability class ξ. Since multiple sub-SFC of a certain reliability class ξ can have n_{c_ξ} components of type c inside a component of type $c-1$ to place VNFs of the same type, the configuration set is noted now as $\epsilon_\xi = \{n_{1_\xi}, n_{2_\xi}, n_{3_\xi}, n_{4_\xi}\}$. For instance, in Fig. 3.11, both sub-SFCs always belong to the same reliability class, if the component reliability of DCs, racks, servers and VMs has equal value, respectively. Considering Pl. 13 and each sub-SFC separately, the configuration sets can be defined as $\epsilon_\xi = \{1, 1, 2, 1\}$. Moreover, both sub-SFCs have same disjointness level of $\Delta_\xi = \Delta_{\xi+1} = 1$ and heterogeneity degree $N_{r_\xi} = N_{r_{\xi+1}} = 2$.

In contrast, if all components of any type c used for placement of all n sub-SFCs have the same component dependent reliability, i.e., all utilized components from the same hierarchy level have the same component dependent reliability, but each sub-SFC follows a different placement strategy, the sub-SFCs still belong to N different reliability classes, where $1 \leq N \leq n$. That is because each sub-SFC is placed in a different fashion involving a different number of DCN components as well as has a level of disjointness Δ_ξ and a heterogeneity degree N_{r_ξ}, which is different from all others. As shown in Fig. 3.12, all sub-SFCs are placed differently, i.e., their levels of disjointness and heterogeneity degrees are different. Here, the heterogeneity degree N_r can not be explicitly defined. Thus, the sub-SFCs can not be combined into $N < n$ reliability classes.

As can be seen in Fig. 3.12, some sub-SFCs from different reliability classes utilize the same components, i.e., R1 and DC7. We refer to

components of any hierarchy level jointly utilized by sub-SFCs from different reliability classes as the **common root components**. For instance, the sub-SFCs of reliability class $\xi = 1, 2$ have two common roots, i.e., R1 and DC7. Obviously, that any common root is a special shared component, i.e., a failure of any common root results in the failure of multiple different reliability classes, i.e., whole sub-SFCs. Moreover, DC7 is the common root for three reliability classes, i.e., $\xi = 1, 2, 3$, while the sub-SFC of class $\xi = 4$ is placed separately from other reliability classes and does not have any common root components with them.

To describe common root components of any type c and all related reliability classes combined by the root component c, we introduce a set $\Phi = \{c_{w_1}, c_{w_2}, ..., c_{w_\rho}, ...\}$, where any c_{w_ρ} describes a component type of common root, each w_ρ presents a set of reliability classes combined by the root component c_{w_ρ}, i.e., $w_\rho = \{\xi_1, ..., \xi_\rho, ...\}$, where $c_{\xi_1} \equiv ... \equiv c_{\xi_\rho} \equiv ... \equiv c_{w_\rho}$. The variable ρ is an index which shows the order of any element in a certain set. Since all components of type $\mathcal{C}$ are always disjoint, i.e., use different VMs, they can not be the common root components c_{w_ρ} resulting in $1 \leq c_{w_\rho} \leq \mathcal{C} - 1$, i.e., DC, rack or server. Considering the example from Fig. 3.12, the configuration set for the common roots can be determined as $\Phi = \{2_{w_1}, 1_{w_2}\}$ with $w_1 = \{1, 2\}$ and $w_2 = \{1, 2, 3\}$.

Note that using the component type $\mathcal{C}-1$, i.e., server, as a common root component will result in the same placement strategies for all sub-SFCs, which utilize this root. Thus, when all components of type $\mathcal{C}$ have the same component dependent reliability, i.e., $p_{\mathcal{C}_1} = ... = p_{\mathcal{C}_\xi} = p_{\mathcal{C}_{\xi+1}} = ...$, all sub-SFCs connected by the common root $\mathcal{C} - 1$ will be from the same reliability class and, thus, the common root can be considered as a shared component only.

3.4.6 Combinatorial Analysis of SFC Reliability

The analytical models for availability and reliability are generally interchangeable and only defined through availability or reliability of the individual components [26]. Thus, we define SFC reliability as a probability that at least $k \geq 1$ sub-flows can successfully traverse k out of $n = k + r$ sub-SFCs to complete the requested service, where $r \geq 0$. Since each sub-SFC consists of Ψ different VNF types, at least $k\Psi$ VNFs of each type have to be available during service run-time. Our reliability analysis assumes that all components of type c that host Ψ VNFs of a certain reliability class ξ have an equal component dependent reliability as previously discussed.

When backup sub-SFCs are placed so that there are r backup VNFs, any components from any hierarchy level c and any reliability class ξ may fail as long as they result in failure of at most r VNFs of each type. However, there is a need to take into account the VNF placement strategy and the interrelation between involved components of different and same hierarchy levels. To be able to control the number of failed and available components of any type and from any reliability class, which can fail without affecting the overall service maintenance, we introduce a parameter **Acceptable Component Failures (ACF)** A_{c_ξ}. ACF shows the allowed amount of failed hardware and software components that do not lead to interruption of the service. It is obvious that when no backup or other protection is applied, there is no ACF and all active components, i.e., DCs, racks, servers and VNFs, must be available during service run-time for a successful service. Without loss of generality, the ACF strongly depends on the number of backup components and the placement of VNFs of a certain reliability class inside DCs, racks and servers. Additionally, we introduce a variable Λ_{c_ξ}, which describes the **number of remaining available components** of a type c, which were not affected by occurred failures of any other components, but can still fail during service run-time. Using these parameters, we next

derive the placement independent and placement dependent end-to-end service, i.e., SFC, reliability, which is generally the probability that at least $\Lambda_c - A_c$ out of Λ_c components of any type c are available and, thus, described by a well known Binomial formula determined as $\sum_{f_c=0}^{A_c} p(\Lambda_c, f_c, p_c) = \sum_{f_c=0}^{A_c} \binom{\Lambda_c}{f_c} p_c^{\Lambda_c - f_c}(1-p_c)^{f_c}$, where $p(\Lambda_c, f_c, p_c)$ is a probability mass function of binomial distribution.

Placement Independent SFC Reliability

In DCNs, where only components of the lowest hierarchy level $c = \mathcal{C} = 4$, i.e., VNFs, can fail and components of hierarchy levels from 1 to $\mathcal{C}-1 = 3$, i.e., DCs, racks and servers, have the reliability of 100%, i.e., $p_1 = p_2 = p_3 = 1$ for any reliability class ξ, the SFC reliability is independent of VNF placement strategy. Then, the end-to-end service is successful if at least k out of n VNFs from any out of N reliability classes are available. Without loss of generality, two sub-SFCs belong to two different reliability classes ξ and $\xi + 1$, if they utilize VNFs with a different component dependent reliability, i.e., $p_{4_\xi} \neq p_{4_{\xi+1}}$. Each reliability class ξ, however, can include $n_\xi \leq n$ sub-SFCs and, thus, n_ξ VNFs of any type, i.e., $\sum_{\xi=1}^{N} n_\xi = n$.

Lemma 3.1 *The placement independent SFC reliability is only a function of the number of reliability classes N, SFC length Ψ, ACF A_{4_ξ} and the amount of available VNFs Λ_{4_ξ} of any type and any reliability class ξ and can be determined as follows*

$$R(N) = \left[\prod_{\xi=1}^{N} \sum_{f_{4_\xi}=0}^{A_{4_\xi}} p(\Lambda_{4_\xi}, f_{4_\xi}, p_{4_\xi}) \right]^{\Psi}, \tag{3.19}$$

where $A_{4_\xi} = min\{n_\xi, r - \sum_{\xi=1}^{\xi-1} f_{4_\xi}\}$ and $\Lambda_{4_\xi} = n_\xi$.

The proof of Lemma 3.1 is presented in Appendix 7.5.

Placement Dependent SFC Reliability

When any DCN component, i.e., DC, rack, server and VNF, can fail, i.e., $p_1 < 1$, $p_2 < 1$, $p_3 < 1$ and $p_4 < 1$, the SFC reliability is a function of the amount of available and failed active and backup DCN components and their interdependency determined by selected VNF placement strategy over Intra- and Inter-DCN. Let us further simplify notation as $P(\Lambda_c) = p(\Lambda_c, f_c, p_c)$ to reduce the size of some formulas provided below, where $1 \leq c \leq \mathcal{C}$, $c \in \{4, 3, 2, 1\}$. Here, the assumption that all components from a hierarchical level c, which host the sub-SFCs of reliability class ξ, have the same component dependent reliability p_{c_ξ} is still valid. Since any reliability class ξ utilizes n_{1_ξ} DCs, n_{2_ξ}, n_{3_ξ} servers inside any rack and n_{4_ξ} VNFs inside any server, all n_{1_ξ} DCs, all $(n_{1_\xi} \cdot n_{2_\xi})$ racks, all $(n_{1_\xi} \cdot n_{2_\xi} \cdot n_{3_\xi})$ servers and all Ψ VNFs have the same component dependent reliability p_{1_ξ}, p_{2_ξ}, p_{3_ξ} and p_{4_ξ}, respectively.

First, a reliability analysis of SFC, which consists of n sub-SFCs and belongs to a single reliability class, is presented. There can be only one reliability class $N = 1$ when all n active and backup sub-SFCs are placed in the same fashion, and all components of each type c have the same component dependent reliability, i.e., p_c. To accommodate all n VNFs of a certain type, e.g., VNF1, there is a need for $1 \leq n_{1_1} \leq n$ active and backup DCs, $1 \leq n_{2_1} \leq n$ active or backup racks inside any DC, i.e., in total $n_{1_1} n_{2_1}$ active and backup racks, and $1 \leq n_{3_1} \leq n$ active or backup servers inside any rack, i.e., in total $n_{1_1} n_{2_1} n_{3_1}$ active and backup servers, and $(k + r)$ VMs. Here, any server can host $1 \leq n_{4_1} \leq n$ VNFs of the same type. As there is only one reliability class considered, let us first simplify notation n_{c_ξ} and p_{c_ξ} as n_c and p_c, respectively. Similarly, the same number of DCs, racks and servers is utilized to place each other VNF type from the same sub-SFC, i.e., VNF2, ..., VNFΨ. Thus, the number of components from hierarchy level c used to place a SFC can be determined as n_c, if $c = 1$, and as $n_1 \prod_{c'=2}^{c} n_{c'}$, if

$c > 1$. Some of utilized components, i.e., Δ different component types from the lower hierarchy level, can host a certain VNF type only, other $\mathcal{C} - \Delta$ component types are the shared components and host VNFs of different types. Thus, there is a need to consider the VNF placement strategy and the resulting level of disjointness Δ, i.e., the components' interdependency.

Lemma 3.2 *When there is one reliability class only, $N = 1$, the placement dependent SFC reliability is a function of the disjointness level Δ, SFC length Ψ, ACF A_c and the number of available components Λ_c for any component type c, i.e.,*

$$R_{\Delta} = \prod_{c=1}^{\mathcal{C}-\Delta} \sum_{f_c=0}^{A_c} P(\Lambda_c) \left[\prod_{c=\mathcal{C}-\Delta+1}^{\mathcal{C}} \sum_{f_c=0}^{A_c} P(\Lambda_c) \right]^{\Psi}, \tag{3.20}$$

where $\prod_{c=1}^{\mathcal{C}-\Delta} \sum_{f_c=0}^{A_c} P(\Lambda_c) = 1$, if $\Delta = \mathcal{C}$.

The proof of Lemma 3.2 is presented in Appendix 7.6.

Now, we consider the parallelized SFC, whereby any out of n active and backup sub-SFCs can belong to one out of N, where $1 \leq N \leq n$, reliability classes. An index ξ, $1 \leq \xi \leq N$, shows a certain reliability class ξ of any active or backup component utilized to place the sub-SFCs belonging to this reliability class. Thus, any component type c involved in placement of reliability class ξ and its component reliability are noted as c_{ξ} and $p_{c_{\xi}}$, respectively. Multiple sub-SFCs can belong to a certain reliability class ξ if these sub-SFCs are placed following a certain placement strategy and all $\mathcal{C}$ component types involved have a certain value of component reliability, i.e., $p_{c_{\xi}}$. It is assumed that all components of hierarchy level c which accommodate any Ψ different VNFs of a certain SFC reliability class ξ have the same component dependent reliability, i.e., $p_{c_{\xi}^1} = ... = p_{c_{\xi}^{\psi}} = ... = p_{c_{\xi}^{\Psi}} = p_{c_{\xi}}$, where a variable ψ, $1 \leq \psi \leq \Psi$, shows a VNF type. Generally, there can be $\Psi \geq 1$ VNFs per sub-SFC, any number of component types

($\mathcal{C}$ types) involved to place these VNFs and any number of reliability classes (N), whereby each reliability class has its own level of disjointness, $1 \leq \Delta_\xi \leq \mathcal{C}$. The level of disjointness Δ_ξ takes into account a certain placement strategy and describes the amount of disjoint and shared components. Additionally, some sub-SFCs from different reliability classes can have common root components, e.g., are allocated to the same DC or the same rack. The failure of these common root components results in failures of multiple sub-SFCs and all related component types of lower hierarchy levels, which belong to different reliability classes. All reliability classes combined by the common root and the related root components are collected in set $\Phi = \{c_{w_1}, c_{w_2}, ..., c_{w_\rho}, ...\}$, where any c_{w_ρ} describes a component type of common root, and each w_ρ is a set of indexes of the reliability classes combined by the common root component of type c_{w_ρ}, i.e., $w_\rho = \{\xi_1, \xi_2, ...\}$ and $c_{\xi_1} \equiv c_{\xi_2} \equiv ... \equiv c_{w_\rho}$. Then, the resulting formula for SFC reliability needs to take into account $N \geq 1$ reliability classes, all $\Psi \geq 1$ VNF types, the shared and common root components utilized for placement of any reliability class ξ.

Lemma 3.3 *When there are multiple reliability classes, $N \geq 1$, the placement dependent SFC reliability is a function of the disjointness level Δ, SFC length Ψ, ACF A_c and the number of available components Λ_c for any component type c. Here, the shared and common root components utilized for placement of sub-SFCs of a reliability class ξ need to be taken under consideration, i.e.,*

$$R_{\Delta_1,...,\Delta_N}(\Phi) = \prod_{\rho=1}^{|\Phi|} \sum_{f_{c_{w_\rho}}=0}^{A_{c_{w_\rho}}} P(\Lambda_{c_{w_\rho}}) \cdot \cdot \prod_{\xi=1}^{N} \prod_{\substack{c_\xi=1 \\ \phi(c_\xi)=1}}^{\mathcal{C}_\xi - \Delta_\xi} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi}) \left[\prod_{\xi=1}^{N} \prod_{\substack{c_\xi=\mathcal{C}_\xi - \Delta_\xi + 1 \\ \phi(c_\xi)=1}}^{\mathcal{C}_\xi} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi}) \right]^{\Psi}, \tag{3.21}$$

where the function $\phi(c_\xi)$ ensures that the common root components

for any reliability class $\xi \in w_\rho$ *are considered only once by the first summation over* $f_{c_{w_\rho}} \equiv f_{c_\xi}$*, i.e.,*

$$\phi(c_\xi) = \begin{cases} 1, & \text{if } \forall \rho : (\xi \in w_\rho \cap c_\xi \neq c_{w_\rho}) \cup (\xi \notin w_\rho), \\ 0, & \text{else.} \end{cases} \tag{3.22}$$

The proof of Lemma 3.3 is presented in Appendix 7.7.

For the reason of comparison between the backup protection and the coding protection, next, a simplified SFC reliability analysis is shown for a use case, where VNFs, servers and path segments can fail, i.e., $\mathcal{C} = 3$ hierarchy levels. DCs and racks are assumed as highly reliable and never fail. For this special case, there are two reliability classes $N = 2$, i.e., the all active and all backup components belong to the first, i.e., $\xi = 1$, and the second, i.e., $\xi = 2$, reliability class, respectively. Moreover, any $m \geq 1$ servers host one sub-SFCs to serve one sub-flow as illustrated in Fig. 3.9, where any active and backup sub-SFC is distributed over $m = 3$ servers in a similar fashion. For instance, the blue sub-SFC of sub-flow f_2 is distributed over S1 and S2 in Rack 1 and S9 in Rack 4, whereby all other sub-SFCs follow the same placement. Then, any server $s \in [1, m]$ utilized by a certain sub-SFC contains $1 \leq \psi_s \leq \Psi$, $\sum_{s=1}^{m} \psi_s = \Psi$, different VNF types. In Fig. 3.9, $\psi_1 = 3$ VNFs, $\psi_2 = \psi_3 = 1$, i.e., $\Psi = 5$. Another assumption is that a connection, i.e., a path segment, to a certain main or backup server out of all alternative connections to the same server exists with a probability of connectivity p_{1_1} or p_{1_2}, respectively. The probabilities of connectivity are the same for all main and all backup servers, respectively. As a result, for $\psi_s > 1$, all m servers ($c = 2$) and m path segments ($c = 1$) are shared components for each related sub-SFC resulting in disjointness level $\Delta_\xi = 1$. Since any sub-SFC is distributed over m servers, at least $(k + r)m$ servers and $k(m+1)$ main- and rm redundant path segments need to be available, whereby k additional path segments provide a connection to the destination, which either decodes or serializes the traffic.

Without any failure protection, all path segments, all servers and VNFs have to be available. Thus, the SFC reliability is defined as

$$R_0(k) = \left[p_{1_1}^{m+1} p_{2_1}^{m} \prod_{s=1}^{m} p_{3_1}^{\psi_s}\right]^k. \tag{3.23}$$

To provide a certain level of SFC reliability, i.e., protection of k active sub-SFCs, there are r backup sub-SFCs allocated to rm backup servers. In case of failures, the backup VNFs can replace failed active VNFs of the same type. Thus, all active and backup servers in DCN need to be connected by at least one available path segment. The path segments from the last server, i.e., $s = m$, to the destination, i.e., end-host within DCN, have to be always available with probability $p_{1_1}^k$ and, thus, represent an unprotected shared component. In contrast, a failure of any other path segment to active or backup servers results in traffic redirection to a reachable backup server. In this case, the path segment to a certain backup server needs to be available. In other words, when a server can not be reached due to connectivity failure, the reachable backup server replaces this unreachable server. Considering the special placement strategy with nm servers and path segments involved, the SFC reliability with backup VNF protection can be derived with Eq. (3.21) for $R_{\Delta_1=\Delta_N=1}(\emptyset)$ as $R_b = p_{1_1}^k \prod_{s=1}^{m} R_{\Delta_1=\Delta_N=1}(\emptyset)$ resulting in

$$R_b = p_{1_1}^k \prod_{s=1}^{m} \prod_{\xi=1}^{N} \prod_{c_\xi=1}^{\mathcal{C}_\xi - 1} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi}) \left[\prod_{\xi=1}^{N} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi})\right]^{\psi_s}. \tag{3.24}$$

Acceptable Component Failure

The bound of summations $A_{c_{w_\rho}}$ and A_{c_ξ} in Eq. (3.21) is what we refer to as acceptable component failure. ACF is a function of pre-reserved components of type c from the same reliability class ξ and the failures of any other components of higher hierarchy level over all reliability classes or failures of components of the lowest

hierarchy level of any reliability class ξ', where $\xi' < \xi$. Depending on the placement strategy and, thus, the reliability class ξ, there are n_{1_ξ} active and backup components of the highest hierarchy level such as DCs and n_{c_ξ} components of any type $c > 1$ placed inside a component of type $c-1$. Thus, there are in total $n_{1_\xi} \prod_{c'=2}^{c} n_{c'_\xi}$ components of type c utilized for placement of n_ξ active and backup VNFs of certain type and any reliability class ξ.

Without loss of generality, each summation in Eq. (3.21) is a function of the prior summation, i.e., any A_{c_ξ} and Λ_{c_ξ} are the functions of already considered component failures $A_{c_\xi} \equiv A_{c_\xi}(f_{1_1}, ..., f_{c_1}, ..., f_{\mathcal{C}_1}, ..., f_{1_\xi}, ..., f_{c_\xi}, ..., f_{\mathcal{C}_\xi}, ...)$ and $\Lambda_{c_\xi} \equiv \Lambda_{c_\xi}(f_{1_1}, ..., f_{c_1}, ..., f_{\mathcal{C}_1}, ..., f_{1_\xi}, ..., f_{c_\xi}, ...)$, where the previously assumed failures reduce the acceptable failures of component c_ξ and the amount of remaining available components of type c from class ξ, respectively, where $1 \leq \xi \leq N$. Next, we derive the amount of available components Λ_{c_ξ} and ACF A_{c_ξ}, where $1 \leq \xi \leq N$ and $1 \leq c \leq \mathcal{C}$.

Let us first derive ACF $A_{c_{w_\rho}}$ of the common root components of any reliability class ξ from set w_ρ, i.e., $\forall \xi : \xi \in w_\rho$. Since there is only one common root component of $|w_\rho|$ reliability classes, the number of available roots $\Lambda_{w_\rho} \leq 1$ and the ACF of the root component is either 0 or 1. Both values, Λ_{w_ρ} and $A_{c_{w_\rho}}$, depend on the number of failed VNFs of each type in the case of a certain root failure and failed roots from a higher hierarchy level. Thus, the failure of a certain common root c_{w_ρ} is only possible, when the other roots from the higher hierarchy do not fail leading to the failure of this root c_{w_ρ}, i.e., $\forall \rho', 1 \leq \rho' \leq \rho - 1 : f_{c_{w_\rho}} \leq 1$, if $f_{c_{w_{\rho'}}} = 0 \cap w_\rho \subset w_{\rho'}$. In other words, the root component can only fail if it is available, i.e., not failed due to failures of other components, $\forall \rho', 1 \leq \rho' \leq \rho - 1 : \Lambda_{c_{w_\rho}} = 1$, if $f_{c_{w_{\rho'}}} = 0 \cap w_\rho \subset w_{\rho'}$. As a result, the amount of

available common root components can be determined as follows

$$\Lambda_{c_{w_\rho}} = \begin{cases} 1, \text{ if } \sum\limits_{\substack{\rho'=1 \\ w_\rho \subset w_{\rho'}}}^{\rho-1} f_{c_{w_{\rho'}}} = 0, \\ 0, \text{ else.} \end{cases} \tag{3.25}$$

Generally, to maintain the service, the amount of failed VNFs may not be larger than the number of backup VNFs, i.e., r. The amount of VNFs of a certain type of reliability class $\xi \in w_\rho$ can be calculated as $n_{c_{w_\rho}} \prod_{c=c_{w_\rho}+1}^{\mathcal{C}} n_{c_\xi} = \prod_{c=c_{w_\rho}+1}^{\mathcal{C}} n_{c_\xi}$ VNFs of class ξ, which will fail with a failure of the root component c_{w_ρ}. Since there are $|w_\rho|$ reliability classes affected by failure of root component c_{w_ρ}, the total number of failed VNFs of certain type from any reliability class $\xi \in w_\rho$ is determined as $\sum_{i=1}^{|w_\rho|} \prod_{c=c_{w_\rho}+1}^{\mathcal{C}} n_{c_{\xi_i}}$, where $\xi_i \in w_\rho$ and $\xi_i \equiv \xi$. Since there can be other common roots from the higher hierarchy level which affect the considered root and other roots as well, there is a need to consider all other VNF failures due to failure of all related $\rho - 1$ roots. Thus, the amount of failed VNFs of certain type over ρ root components can be calculated as a function of any $f_{c_{w_\rho}}$, i.e., $\sum_{\rho'=1}^{\rho} f_{c_{w_{\rho'}}} \sum_{\substack{i=1 \\ \sigma=1}}^{|w_{\rho'}|} \prod_{c=c_{w_{\rho'}}+1}^{\mathcal{C}} n_{c_{\xi_i}}$, where $\sigma = 1$, iff $\forall \rho'', 1 \leq \rho'' \leq \rho' - 1 : \xi_i \notin w_{\rho''}$, and ensures that the failures of VNFs from the same reliability class are considered only once. The formula above takes into account placement strategies, where, for instance, DC and rack inside this DC represent two common roots for the same reliability classes, whereby DC can combine more reliability classes and lead to failure of all racks and all reliability classes. Summarizing all constraints above, the ACF of the common root components is

$$A_{c_{w_\rho}} = \begin{cases} 1, \text{ if } (r \geq \sum\limits_{\rho'=1}^{\rho} f_{c_{w_{\rho'}}} \sum\limits_{\substack{i=1 \\ \sigma=1}}^{|w_{\rho'}|} \prod\limits_{c=c_{w_{\rho'}}+1}^{\mathcal{C}} n_{c_{\xi_i}}) \cap \Lambda_{c_{w_\rho}} = 1, \\ 0, \text{ else,} \end{cases} \tag{3.26}$$

where $\sigma = 1$, iff $\forall \rho'', 1 \leq \rho'' \leq \rho' - 1 : \xi_i \notin w_{\rho''}$.

The amount of available active and backup components Λ_{c_ξ} of any component type c_ξ for any reliability class ξ, after the failure of any component of higher hierarchy level, i.e., of type 1 to $c-1$, is

$$\Lambda_{c_\xi}=\begin{cases}\Lambda_{c_{w_\rho}}, & \text{if } \forall\rho:(\xi\in w_\rho)\cap(c_\xi=c_{w_\rho}),\\ n_{\mathcal{C}_\xi}, & \text{if } \forall\rho:(\xi\notin w_\rho)\cap(c=1),\\ (\Lambda_{(c-1)_\xi}-f_{(c-1)_\xi})n_{c_\xi}, & \text{if } \forall\rho:(\xi\notin w_\rho)\cap(1<c\leq\mathcal{C}),\end{cases} \tag{3.27}$$

where the first case determined as $n_{1_\xi}=n_{c_\xi}=n_{c_{w_\rho}}=\Lambda_{c_{w_\rho}}$, describes a number of the common root components of a certain type c and is the same for any reliability class ξ from set w_ρ. However, when there is one reliability class only, $N=1$, there are no common root components and the first case in Eq. (3.27) will be never true. The derivation of Eq. (3.27) is presented in Appendix 7.8.

Generally, the number of any failed components f_{c_ξ} of type c_ξ can vary as $0\leq f_{c_\xi}\leq A_{c_\xi}$ for shared components and $0\leq f_{c_\xi}\leq A'_{c_\xi}$ for disjoint components from any certain reliability class ξ, whereby any f_{c_ξ} out of Λ_{c_ξ} components of type c_ξ of reliability class ξ can fail without interrupting the end-to-end service. Without loss of generality, ACF, i.e., A_{c_ξ} for components shared by Ψ different VNF types of class ξ and A'_{c_ξ} for disjoint components of the same class, is a function of available components Λ_{c_ξ} of type c and a reliability class ξ, the amount of provided backup VNFs r, and the number of VNFs considered as already failed F_{c_ξ} after f_{c_ξ} components of any type c, $1\leq c\leq\mathcal{C}$, and any class ξ, $1\leq\xi\leq\mathcal{C}$ failed. The amount of failed components F_{c_ξ} of the lowest hierarchy level $\mathcal{C}$ and a certain type, e.g., VNFs1, due to failures ($f_{c'_\xi}$) of any component types c', i.e., $c'\leq c$ belonging to a certain reliability class ξ can be defined as

$$F_{c_\xi}=\begin{cases}F_{(c-1)_\xi}+f_{c_\xi}\prod\limits_{c'=c+1}^{\mathcal{C}}n_{c'_\xi} & \text{, if } 1\leq c\leq\mathcal{C},\\ 0 & \text{, else,}\end{cases} \tag{3.28}$$

where the failure of any component from the higher hierarchy level, from 1 to c, is taken into account. These F_{c_ξ} failed components of

type $\mathcal{C}$ due to failures of f_{c_ξ} components of the type from 1 to c reduce the number of available backup components, which are allowed to fail, i.e., reduces ACF. A failure of any common root c_{w_ρ} is determined by Eq. (3.28) as $\forall\rho : f_{c_{w_\rho}} \equiv f_{c_\xi}$, if $c_{w_\rho} = c_\xi$ and $\xi \in w_\rho$.

Then, ACF A_{c_ξ} of any shared component, i.e., placed outside the brackets in Eq. (3.21), is a minimum between the total number of available VNFs provided by a reliability class ξ and placed inside component c, i.e., $\Lambda_{c_\xi} \prod_{c'=c_\xi+1}^{\mathcal{C}} n_{c'_\xi}$ and the total number of available backup VNFs r reduced by any failures of any component type and from any reliability class. The result will show how many VNFs of a reliability class ξ may fail without any effect of service maintenance. The amount of backup VNFs is generally reduced by the VNF failures of class ξ due to failures of the components from the higher hierarchy level, i.e., from 1 to $c-1$, and from the same reliability class ξ, which can be calculated with Eq. (3.28) as $F_{(c-1)_\xi}$. Additionally, the number of backup VNFs is reduced through VNF failures due to the failure of any shared component from other reliability classes with an index which is less than ξ. That failures can be also determined with Eq. (3.28) as $\sum_{l=1}^{\xi-1} F_{(\mathcal{C}-\Delta_l)_l}$, where $\mathcal{C}-\Delta_l$ defines the lowest hierarchy of a shared component related to the reliability class l. Thus, defining ACF for component c_ξ, we need to take into account that there are $r - \sum_{l=1}^{\xi-1} F_{(\mathcal{C}-\Delta_l)_l} - F_{(c-1)_\xi}$ remaining backup VNFs, which can fail without service interruption. Thus, the amount of VNFs from class ξ, which may fail are defined as $min\{\Lambda_{c_\xi} \prod_{c'=c_\xi+1}^{\mathcal{C}} n_{c'_\xi}; r - \sum_{l=1}^{\xi-1} F_{(\mathcal{C}-\Delta_l)_l} - F_{(c-1)_\xi}\}$. Since we are interested in ACF of component c_ξ, which can be from any hierarchy level, i.e., $1 \leq c \leq \mathcal{C}$, there is a need for the normalization by the amount of VNFs hosted by component c_ξ, i.e., by $\prod_{c'=c+1}^{\mathcal{C}} n_{c'_\xi}$. That is also the number of VNFs, which will fail if one component c_ξ fails. As a result, ACF for a component type c_ξ shared by Ψ

different VNFs of a reliability class ξ can be derived as

$$A_{c_\xi} = \left\lfloor \frac{min\left\{ \Lambda_{c_\xi} \prod_{c'=c+1}^{\mathcal{C}} n_{c'_\xi}; r - \sum_{l=1}^{\xi-1} F_{(\mathcal{C}-\Delta_l)_l} - F_{(c-1)_\xi} \right\}}{\prod_{c'=c+1}^{\mathcal{C}} n_{c'_\xi}} \right\rfloor, \tag{3.29}$$

where the sum of all $F_{(\mathcal{C}-\Delta_l)_l}$ identifies all summations in Eq. (3.21) located before sum with a bound A_{c_ξ} and, finally, all prior failures of shared components. Obviously, that if any prior reliability class $l < \xi$ have a disjointness level of $\Delta_l = \mathcal{C}$, all Ψ VNFs as well as all related component types of this class l are disjoint and there are no shared components and, thus summations outside the brackets for this reliability class in Eq. (3.21), i.e., $F(\Delta_l) = 0$. When $\Delta_l < \mathcal{C}$, a failure of $\mathcal{C} - \Delta_l$ components types of a certain class l needs to be considered for calculation of ACF A_{c_ξ}, where $l < \xi$. Thus, the function $F_{(\Delta_\xi - 1)_\xi}$ provides a total number of the failed components of the lowest hierarchy level due to the failure of any component types from up to $\Delta_\xi - 1$ hierarchy level.

Similar, ACF for any disjoint, i.e., unshared, component type c_ξ and any reliability class ξ, i.e., A'_{c_ξ} for summations inside the brackets in Eq. (3.21), is a function of the remaining available components Λ_{c_ξ} of type c from a reliability class ξ, the amount of provided backup VNFs r and the number of VNFs considered as failed.

Thus, there is a need to define the amount of VNFs, which may still fail without interrupting the service. Without loss of generality, that is the minimum between the total number of available VNFs provided by the reliability class ξ and placed inside component c_ξ ($\Lambda_{c_\xi} \prod_{c'=1}^{c-1} n_{c'_\xi}$), and the amount of remaining backup VNFs after the failure of any other components from any other reliability classes. Defining ACF of any disjoint component type c_ξ, the number of backup VNFs r is already reduced by possible failures of components from hierarchy level 1 to $\mathcal{C}$ of all reliability classes l, $1 \leq l \leq \xi - 1$. These failures can be calculated with Eq. (3.28) as $\sum_{l=1}^{\xi-1} F_{\mathcal{C}_l}$ failed VNFs from $\xi - 1$ different reliability classes. Some VNFs from

reliability class ξ could fail due to a failure of components from higher hierarchy level belonging to the same class ξ. The number of failed VNFs from reliability class ξ is calculated with Eq. (3.28) as $F_{(c-1)_\xi}$, which additionally reduce the number of available backup VNFs. Moreover, the shared components of any reliability class l, $\xi < l \leq N$, i.e., these reliability classes are not yet considered inside the brackets in Eq. (3.21), can fail resulting in $\sum_{l=\xi+1}^{N} F_{(\mathcal{C}-\Delta_l)_l}$ additional VNF failures. Thus, the amount of backup VNFs of a certain type is determined as $r - \sum_{l=1}^{\xi-1} F_{\mathcal{C}_l} - F_{(c-1)_\xi} - \sum_{l=\xi+1}^{N} F_{(\mathcal{C}-\Delta_l)_l}$ and defines the maximal number of VNFs from the reliability class ξ which may fail due to failures of component c_ξ, which can be reduced through value $\Lambda_{c_\xi} \prod_{c'=1}^{c-1} n_{c'_\xi}$. To calculate the amount of components c_ξ, which may fail resulting in allowed VNFs failures calculated above, the required normalization is performed similar to Eq. (3.29) as $\prod_{c'=c+1}^{\mathcal{C}} n_{c'_\xi}$ to determine the number of VNFs placed inside component c_ξ. As a result, ACF for a component type c_ξ utilized for a disjoint placement of Ψ different VNFs of a reliability class ξ can be calculated as follows

$$A'_{c_\xi} = \left\lfloor \frac{min\left\{ \Lambda_{c_\xi} \prod_{c'=1}^{c-1} n_{c'_\xi}; r - \sum_{l=1}^{\xi-1} F_{\mathcal{C}_l} - F_{(c-1)_\xi} - \sum_{l=\xi+1}^{N} F_{(\mathcal{C}-\Delta_l)_l} \right\}}{\prod_{c'=c+1}^{\mathcal{C}} n_{c'_\xi}} \right\rfloor . \quad (3.30)$$

Reliability Analysis of Coded SFC

To compare coding protection with backup protection, we only consider three types of failures, i.e., failures of servers, VNFs and connections between servers (path segments). Thus, DCs and racks are assumed as highly reliable and never fail. That results in $\mathcal{C} = 3$ hierarchy levels. For this use case, we define two reliability classes $N = 2$, i.e., the active and redundant components belong to the class $\xi = 1$ and $\xi = 2$, respectively. Here, any $m \geq 1$ servers host one sub-SFCs to serve one sub-flow as illustrated in Fig. 3.9, whereby all other sub-SFCs follow the same placement. Then, any server

$s \in [1, m]$ can contain $1 \leq \psi_s \leq \Psi$ different VNF types. Any path segment to the certain main or redundant server is available with a probability of connectivity p_{1_1} or p_{1_2}, whereby these probabilities are the same for all main and all redundant servers, respectively. Moreover, there is a need for k additional path segments to provide a connection to the decoder. To protect k main sub-SFCs, there are r redundant sub-SFCs placed in rm redundant servers. The redundant sub-SFCs are processing the coded redundant sub-flows. In contrast to backup protection, the connectivity to destination within DCN (decoder) can fail with probability $(1 - p_{1_1})$ or $(1 - p_{1_2})$ without degrading service success. Any component failure will result in acceptable traffic loss, whereby any coded redundant sub-flow can replace any other lost sub-flow, which is then recovered by decoding.

To analyze the reliability of a parallelized SFC protected by coding, there is a need to take into account two different types of VNFs, i.e., h-VNF and p-VNF. The coding protection can be applied in the case of header processing only, i.e., SFC consists of h-VNFs. Then, the SFC reliability R_h or $R_h^{'}$ of any main or redundant sub-flow is generally independent of other parallel coded sub-flows and is only the function of connectivity, server and VNF reliability, i.e.,

$$R_h = p_{2_1}^m p_{3_1}^{\Psi} p_{1_1}^{m+1} \text{ or } R_h^{'} = p_{2_2}^m p_{3_2}^{\Psi} p_{1_2}^{m+1}. \tag{3.31}$$

Without coding and VNF redundancy, the SFC reliability can be determined by Eq. (3.23). When r redundant sub-flows are generated by encoding and sent over r parallel redundant paths through r parallel redundant sub-SFCs, the SFC reliability can be calculated with Eq. (3.32) as a probability that at least k sub-flows out of $(k + r)$ parallel coded sub-flows can reach the decoder:

$$R_c = \sum_{i=0}^{r} \binom{k}{i} R_h^{k-i} (1 - R_h)^i \sum_{j=0}^{r-i} \binom{r}{j} {R_h^{'}}^{r-j} (1 - R_h^{'})^j, \tag{3.32}$$

where the first sum considers a possible loss of main sub-flows and the second sum describes the possible acceptable loss of the remain-

ing redundant sub-flows. Both losses, however, are the result of server or VNF failure or connectivity loss regarding Eq. (3.31).

In the case of payload processing, the decoding is only possible if k out of $k + r$ sub-flows are not interrupted and reach the decoder. Thus, the probability of service success depends on both R_c and R_b. Let us assume that any SFC can be split into multiple parts, i.e., M_h chained parts, where $\Psi_{\mathrm{h},i}$, $1 \leq i \leq M_\mathrm{h}$, h-VNFs can process coded packets (GCH), and M_p chained parts with $\Psi_{\mathrm{p},j}$, $1 \leq j \leq M_\mathrm{p}$ p-VNFs, which require prior decoding and backup protection. Then, the total number of different VNFs in any SFC is $\Psi = \sum_{i=1}^{M_\mathrm{h}} \Psi_{\mathrm{h},i} + \sum_{j=1}^{M_\mathrm{p}} \Psi_{\mathrm{p},j}$. The probability of service success in case of hybrid protection can be derived with Eqs. (3.24) and (3.32) based on the serial reliability model as $R = R_b^{M_\mathrm{p}} \cdot R_c^{M_\mathrm{h}}$.

Overhead Analysis

For the method of backup protection, the overhead can be considered as a probability of a need to migrate VNFs and to redirect traffic. The overhead of coding protection, however, is defined as the probability of a need for decoding at the destination.

With backup protection, each failed VNF of a certain type can be replaced by any out of r backup VNFs of the same type through redirection of individual sub-flows. Thus, traffic redirection is required if at least one active VNF is not available. Let us derive the probability that all k active sub-SFCs are available as $\prod_{s=1}^{m}(p_{1_1} p_{2_1} p_{3_1}^{\psi_s})^k$, then the probability of traffic redirection is determined as follows

$$P_\mathrm{R} = 1 - \prod_{s=1}^{m}(p_{1_1} p_{2_1} p_{3_1}^{\psi_s})^k. \tag{3.33}$$

Since we apply systematic coding, where k sub-flows are unchanged after encoding, and r sub-flows present a linear combination of the k original sub-flows, the decoding is only required if at least one out of k unchanged sub-flows is lost. There is no need for decoding

when $f \leq r$ redundant sub-flows are lost. Thus, the probability of a decoding need can be derived as follows

$$P_{\text{dec}} = \sum_{f=1}^{r} \sum_{i=0}^{r-f} \binom{k}{f} R_h^{k-f} (1 - R_h)^f \binom{r}{i} R_h^{'\, r-i} (1 - R_h^{'})^i. \quad (3.34)$$

To decrease the probability of decoding need P_{dec}, the k unchanged sub-flows and r redundant modified sub-flows can be sent over more and less reliable routes, respectively.

3.4.7 Performance Evaluation

In this subsection, first, the impact of VNF placement strategies, SFC parallelization, amount of redundant components and the SFC length on the service success is investigated. Then, we compare backup protection and coding protection.

Performance of Backup Protection

First, the placement dependent SFC reliability for a single reliability class ($N = 1$) is evaluated following examples from Fig. 3.11. We investigated all 16 possible placement strategies to symmetrically distribute active and backup VNFs of any type over Inter- and Intra-DCNs. For that reason, failures of DCs, racks, servers and VNFs are considered in two scenarios: 1) Use Case 1 relates to practice, where $p_1 = 0.99999$, $p_2 = 0.9999$, $p_3 = 0.999$, $p_4 = 0.99$; and 2) Use Case 2 is idealized scenario, where $p_1 = p_2 = p_3 = p_4 = 0.99999$. To investigate impact of placement strategies on the SFC reliability, the number of active sub-SFCs was set to $k = 1$, the length of SFC to $\Psi = 4, 8$ VNFs and a number of backup sub-SFCs to $r = 0, 1, 100$. The analytical results calculated with Eq. (3.20) are verified by Monte-Carlo simulations. As the simulation results overlapped with analytical results (with 95% confidence interval), only analytical results are presented in a tabular form for clarity, where N_r and, in most cases, SFC reliability values are sorted in the descending order.

Tab. 3.3 shows the SFC reliability for Use Case 1 and Use Case 2 as a function of the amount of backup sub-SFCs r, SFC length Ψ and different VNF placement strategies defined in Fig. 3.11. Without backup protection ($r = 0$), the SFC reliability is independent of heterogeneity degree N_r. However, as expected, for a fixed N_r, the service success decreases with increasing level of disjointness Δ and increasing SFC length Ψ (for fixed Δ but different $N_r = 1, .., 4$ the reliabilities are identical). As expected, Use Case 2 provides higher SFC reliability due to higher component reliability defined but reflects the same functional dependency. Without any protection, a disjointness of $\Delta = 1$ will provide the highest SFC reliability. This disjointness can be reached with different VNF placement strategies as given by Pl. 11, Pl. 12, Pl. 13 and Pl. 4.

Now, the usual 1+1 backup protection, i.e., $r = 1$, is evaluated in Tab. 3.3. At first glance, the service success decreases with a decreasing amount of backup component types N_r, and an increasing level of disjointness Δ, e.g., for Use Case 1 with $\Psi = 4$. Intuitive, that seems to be a practical rule of thumb. However, some placement strategies with low N_r provide higher service success (marked bold) than some placements with a higher N_r. In Use Case 1 with a SFC length of $\Psi = 8$, Pl. 12 with $N_r = 3$ outperforms Pl. 7, Pl. 5 and Pl. 1 with $N_r = 4$. That is because the SFC in these placement strategies will most likely be interrupted due to VNF failures caused by the longer VNF chain. Then, if we consider the VNF components of a SFC only, its reliability relates to $(p_v)^{\Psi}$, i.e., the lower reliability of the longer SFC chain, i.e., $(p_v)^8 \approx 0.92274 < (p_v)^4 \approx 0.9606$, can not be compensated by the backup DC, R and S. Thus, the amount of other backup (hardware) components, i.e., N_r, can be reduced. Now, Use Case 2 is considered, where the placement strategies based on VNF type are marked bold. As we can observe in Tab. 3.3, a

N_r	Δ	Pl.	Use Case 1						Use Case 2					
			$\Psi = 4$			$\Psi = 8$			$\Psi = 4$			$\Psi = 8$		
			$r = 0$	$r = 1$	$r = 100$	$r = 0$	$r = 1$	$r = 100$	$r = 0$	$r = 1$	$r = 100$	$r = 0$	$r = 1$	$r = 100$
4	1	11	0.9595298551	0.9995123444	$\to 1$	0.9217205502	0.9990295230	$\to 1$	0.9999300030	0.9999999963	$\to 1$	0.9998900055	0.9999999935	$\to 1$
4	2	7	0.9566541431	0.9995075495	$\to 1$	0.9152878303	0.9990157556	$\to 1$	0.9999000045	0.9999999948	$\to 1$	0.9998200153	0.9999999900	$\to 1$
4	3	5	0.9563671756	0.9995073731	$\to 1$	0.9146473210	0.9990150262	$\to 1$	0.9998700078	0.9999999939	$\to 1$	0.9997500300	0.9999999879	$\to 1$
4	4	1	0.9563384848	0.9995073585	$\to 1$	0.9145832976	0.9990149596	$\to 1$	0.9998400120	0.9999999936	$\to 1$	0.9996800496	0.9999999872	$\to 1$
3	1	12	0.9595298551	0.9995031488	0.9999899999	0.9217205502	**0.9990210788**	0.9999899999	0.9999300030	0.9999899976	0.9999899999	0.9998900055	0.9999899956	0.9999899999
3	2	8	0.9566541431	0.9994984114	0.9999899999	0.9152878303	0.9990074399	0.9999899999	0.9999900045	0.9999899967	0.9999899999	0.9998200153	0.9999899935	0.9999899999
3	3	2	0.9563671756	0.9994982407	0.9999899999	0.9146473210	0.9990067233	0.9999899999	0.9998700078	0.9999899964	0.9999899999	0.9997500300	0.9999899928	0.9999899999
3	4	6	0.9563384848	0.9994682561	0.9999900006	0.9145832976	0.9989367949	0.9999200028	0.9998400120	**0.9999599970**	**0.9999600006**	0.9996800496	**0.9999199956**	**0.9999200028**
2	1	13	0.9595298551	0.9994111840	0.9998900009	0.9217205502	0.9989366284	0.9998900009	0.9999300030	0.9999799988	0.9999800001	0.9998900055	0.9999799976	0.9999800001
2	2	3	0.9566541431	0.9994070213	0.9998900009	0.9152878303	0.9989242748	0.9998900009	0.9999900045	0.9999799985	0.9999800001	0.9998200153	0.9999799969	0.9999800001
2	3	10	0.9563671756	0.9991072291	0.9995900640	0.9146473210	0.9982252375	0.9991902879	0.9998700078	**0.9999499994**	**0.9999500010**	0.9997500300	**0.9999100004**	**0.9999100036**
2	4	9	0.9563384848	0.9990772562	0.9995600766	0.9145832976	0.9981553639	0.9991203467	0.9998400120	**0.9999200012**	**0.9999200028**	0.9996800496	**0.9998400088**	**0.9998400120**
1	1	4	0.9595298551	0.9984906149	0.9988901109	0.9217205502	0.9980912785	0.9988911109	0.9999300030	0.9999699999	0.9999700003	0.9998900055	0.9999699995	0.9999700003
1	2	16	0.9566541431	0.9954981375	0.9958964363	0.9152878303	0.9911256646	0.9919188219	0.9999000045	**0.9999400011**	**0.9999400015**	0.9998200153	**0.9999000037**	**0.9999000045**
1	3	15	0.9563671756	0.9951995179	0.9955976973	0.9146473210	0.9904319848	0.9912246871	0.9998700078	**0.9999100032**	**0.9999100036**	0.9997500300	**0.9998300128**	**0.9998300136**
1	4	14	0.9563384848	0.9951696622	0.9955678297	0.9145832976	0.9903626567	0.9911553034	0.9998400120	**0.9998800062**	**0.9998800066**	0.9996800496	**0.9997600268**	**0.9997600276**

Table 3.3: Placement dependent SFC reliability in Use Case 1 and Use Case 2 [5].

placement based on SFC with a lower N_r outperforms the placement strategy based on VNF type, e.g., Pl. 13 outperforms Pl. 6. Moreover, also other placement strategies based on VNF type (marked bold) provide the lowest SFC reliability for any value of r and Ψ. Thus, it is preferable to use a placement strategy based on SFC.

The impact of an increasing number of backup sub-SFCs on service success in both studied use cases is investigated, whereby $r = 100$ can be considered as a good approximation for a reliability upper bound. Tab. 3.3 presents a more detailed performance of SFC reliability for both use cases, which also reflects the case without protection as follows: *For a given value of N_r, the reliability decreases with increasing level of disjointness Δ, and, for a fixed value of Δ, the reliability increases with increasing N_r.* For instance, for a $\Delta = 1$, Pl. 11, Pl. 12, Pl. 13 and Pl. 4 correspond to descending $N_r = 4...1$, and the higher N_r results in the higher SFC reliability for any $r > 0$ and Ψ. Furthermore, the reliability increase with increasing redundancy, i.e., 0.9, 0.999, 1 with $r = 0, 1, 100$, respectively, for Pl.11 in Use Case 1. In contrast, Pl. 13 and Pl.4 with $N_r = 2$ and $N_r = 1$ provide SFC reliability of "3-" and "2-"nines almost independently of value $r \neq 0$, respectively. As discussed above and verified by the results in Tab. 3.3, the highest SFC reliability can be reached by Pl. 11 in any configurations. Here, each SFC is allocated to the same server, i.e., $\Delta = 1$ but VNFs of the same type are placed in different DCs, rack, server and VNFs, i.e., $N_r = 4$. However, for Use Case 1 and Pl. 11, the VNF reliability of $p_v = 0.99$ is too low to guarantee a "5-"nines level, which will only be reached at the impractical bound ($r = 100$). Most important, the placements Pl. 12, Pl. 13 and Pl. 4 can not guaranty the SFC reliability of "5-"nines in both Use Cases, not even as the upper bound with $r = 100$.

Next, the placement strategy Pl.11 with $N_r = 4$ and $\Delta = 1$ is considered only as it provides the highest SFC reliability in setting with $k \geq 1$ active sub-SFCs and Use Case 1 (Tab. 3.3). Tab. 3.4

	$k=1$	$k=4$	$k=8$
$\frac{r}{k}$		$\Psi=4$	
0	0.959529855054096	0.847683965207723	0.718568104870289
0.125	-	-	0.983696719311009
0.25	-	0.995273874413365	0.999384198321856
0.375	-	-	0.999981195954079
0.5	-	0.999893621626573	0.999999500094456
0.625	-	-	0.999999987999969
0.75	-	0.999997932451063	0.999999999734000
0.875	-	-	0.999999999994470
1	0.999512344397461	0.999999963313851	0.999999999999890
1.125	-	-	0.999999999999997
1.25	-	0.999999999389706	0.999999999999999
1.375	-	-	0.999999999999999
$\frac{r}{k}$		$\Psi=8$	
0	0.921720550240843	0.721767099608634	0.520947746077460
0.125	-	-	0.968371471691151
0.25	-	0.990697185787667	0.998774974539633
0.375	-	-	0.999962424513603
0.5	-	0.999787784003278	0.999999000321022
0.625	-	-	0.999999976000402
0.75	-	0.999995866449611	0.999999999468002
0.875	-	-	0.999999999988942
1	0.999029523012867	0.999999926631523	0.999999999999781
1.125	-	-	0.999999999999995
1.25	-	0.999999998779421	0.999999999999998
1.375	-	-	0.999999999999999

Table 3.4: SFC reliability provided by Placement 11 [7].

shows the SFC reliability as a function of k, r and Ψ. The first column contains the amount of utilized backup sub-SFCs r normalized by the number of active sub-SFCs k. The increasing number of VNFs in SFC, i.e., Ψ, decreases the SFC reliability. In contrast, the increasing number of active sub-SFC k significantly increases the SFC reliability with backup protection, $r > 0$. The reliability of "six nines" can be reached with a configuration $k = 4$ and $r = 4$ or $k = 8$ and $r = 4$ for both $\Psi = 4$ and $\Psi = 8$, while serial SFC ($k = 1$) requires $r = 3$ backup sub-SFCs for the same level of SFC reliability, which means 300% redundancy (not shown in Tab. 3.4).

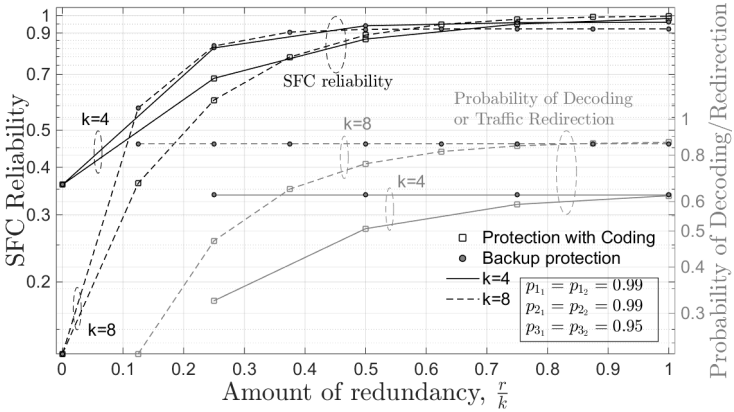


Figure 3.13: SFC reliability and Overhead (Coding or Backup) [6].

Performance of Coding Protection

Now, the performance of coding and backup protections is compared regarding SFC reliability and overhead due to decoding and traffic redirection, respectively. The analytical results were calculated with Eqs. (3.24), (3.32), (3.33) and (3.34). For all results, there are two reliability classes $N = 2$ and $\Psi = 4$ VNFs in any SFC.

Fig. 3.13 shows the SFC reliability and the overhead probability, as a function of the number of main and redundant parallel sub-SFCs, i.e., k and r. In contrast to backup protection, which provides high SFC reliability with a few redundant sub-SFCs, the coding protection outperforms backup protection only if there are $\geq 60\%$ redundant sub-SFCs. Generally, service success can be increased by increasing k and r for both protection methods. The overhead of backup protection, which relates to the need for traffic redirection, is independent of the number of backup sub-SFCs and increases with increasing k. The overhead of coding protection, i.e., the probability of decoding need, is less than the probability of traffic redirection and increases with increasing k and r.

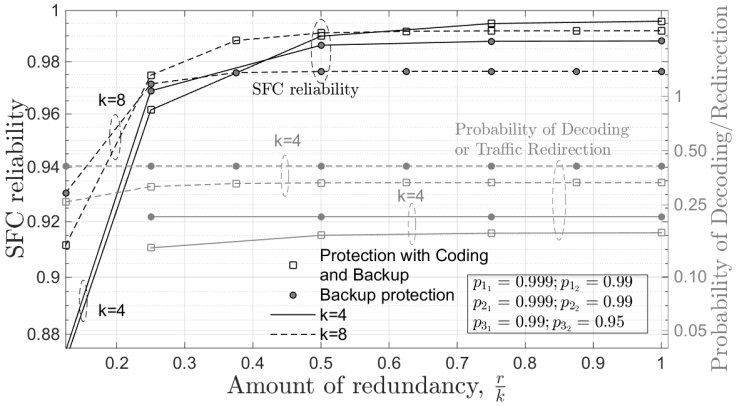


Figure 3.14: SFC reliability and Overhead (Coding with Backup) [6].

Fig. 3.14 shows the SFC reliability and the probability of overhead for the use case presented in Fig. 3.9, where SFC includes both h-VNFs and p-VNFs and requires hybrid backup-coding failure protection. Thus, the amount of redundant and backup VNFs were varied to compare the backup protection with hybrid protection, i.e., coding protection of VNFs1-VNFs3 and VNFs5 and backup protection of VNFs4. The hybrid protection outperforms the pure backup protection for any k providing higher service success and lower probability of decoding need when the amount of redundancy is larger than 25%. When the amount of redundancy is less than ≈ 0.4, the highest SFC reliability can be reached with a high level of flow parallelization $k = 8$. In contrast, the lower level of flow parallelization, i.e., $k = 4$, results in higher SFC reliability when the amount of redundancy is ≥ 0.4. As can be seen in Fig. 3.14, the minimal overhead and the SFC reliability of five nines can be reached with $k = 4$ and hybrid backup-coding protection.

3.4.8 List of Symbols

k	number of original parallel sub-flows and, thus, active sub-SFCs;
r	number of backup sub-SFCs;
n	total number of pre-reserved sub-SFCs, i.e., $n = k + r$;
Ψ	number of VNFs in a SFC and, thus, sub-SFC;
$\mathcal{C}$	number of hierarchy levels, i.e., different component types;
c	certain component type, $1 \leq c \leq \mathcal{C}$;
N	total number of reliability classes, i.e., $1 \leq N \leq n$;
ξ	certain reliability class, $1 \leq \xi \leq N$;
n_ξ	number of SFCs of reliability class ξ, $1 \leq \xi \leq N$
$\bar{n}_{1_\xi}$	number of components from the highest hierarchy of reliability class ξ;
n_{c_ξ}	number of components of type c inside components of type $c-1$;
n_1	number of DC, if $\mathcal{C} = 4$, or a number of path segments, if $\mathcal{C} = 3$;

n_2	number of racks inside a DC, if $\mathcal{C} = 4$, or a number of servers, if $\mathcal{C} = 3$;
n_3	number of servers inside a rack, if $\mathcal{C} = 4$, or a number of VNFs, if $\mathcal{C} = 3$;
n_4	number of VNFs of certain type inside a server, if $\mathcal{C} = 4$;
m	number of servers used to place one sub-SFC, where $m \geq 1$;
ψ_s	number of VNFs allocated to server $s \in [1, m]$;
p_{c_ξ}	component reliability of any component of type c utilized to host class ξ;
p_4	reliability of VNF for $\mathcal{C} = 4$;
p_3	reliability of server for $\mathcal{C} = 4$;
p_2	reliability of rack for $\mathcal{C} = 4$;
p_1	reliability of DC for $\mathcal{C} = 4$;
N_r	number of component types to disjointly place VNFs of the same type;
Δ	number of component types to disjointly place VNFs of different type;

3.5 Summary

This chapter provided a new analytical framework for system engineering and evaluation of the service reliability of advanced network systems with applied diversity, i.e., path and traffic parallelism, coding, and over-provisioning, i.e., component and coding redundancy. First, a reliable parallel FSO transmission system with adaptive coding was designed and analyzed regarding the code rate and its upper bound. The results showed that a reliable parallel FSO transmission is possible even for a high traffic loss when adaptive coding is utilized. Depending on the system requirements, both hybrid RF/FSO parallel transmission and parallel transmission over multiple FSO channels can be designed for highly reliable data transmission, with a trade-off between information rate, complexity, and cost.

Finally, SFC reliability was studied as the reliability of parallelized SFC protected by backup SFCs, systematic erasure coding, or their combination. For backup failure protection, we provided a novel generalized reliability analysis that considers the failure of DCs, racks, servers, and VNFs as well as the component sharing, heterogeneity, and their interdependency. We defined all possible placement strategies over Inter- and Intra-DCNs. We then analytically investigated a placement independent and a placement dependent service reliability as a function of complex failure propagation. The reliability models and analysis presented are based on combinatorics and probability theory. Surprisingly, this simple mathematical approach can be utilized to analyze rather complex and generic SFC configurations. The analytical results confirmed by simulation showed some trade-offs between component heterogeneity and disjointness for SFC placement, whereby the increasing disjointness level does not automatically lead to the highest SFC reliability.

4

Security Analysis of Advanced Network Systems

4.1 Introduction

Security threats such as eavesdropping and jamming can be performed on the physical or IP layer affecting anonymity, privacy, integrity and reliability of transmission. To hamper eavesdropping and jamming, all security tools are required, i.e., randomness, diversity, coding, and over-provisioning. The advanced network systems with the inherent parallel transmission, coding and random routing engineered in Chapter 2 are generally designed to minimize the success of any security attack. However, there is a need to verify the effectiveness of these new network concepts to prevent security attacks and compare their security level with that of the standard systems.

Since optical networks are highly suitable for parallel transmission and can provide randomness and intrinsic diversity of parallel paths, as it was shown in Chapter 2, attention should be paid to the optical layer security. Eavesdropping attacks can be launched by directly accessing the optical channel via fiber tapping [89, 90, 91]. The inter-channel jamming can be launched from a legitimately acquired optical channel [90], whereby the attacker injects a high-power jamming signal into the acquired lightpath inducing severe interference effects and, thus, degrading the signal quality.

In the Internet networks, an adversary can reside in IP routers or

access physical (optical) connections disturbing communication or revealing personal data about users and content of sent messages. Therefore, anonymity networks such as The Onion Routing (Tor) have been developed to provide practical mechanisms assuring user anonymity, privacy, i.e., confidentiality, and mitigating censorship. Today's Tor is built as an overlay network among volunteer systems, where any message exchanged between a pair of Tor onion routers (OR) is encrypted in layers. Anonymous communication is possible through traffic tunneling over a chain of selected ORs (entry, middle, exit). The encryption layers in Tor ensure privacy and client-server unlinkability, which means that an adversary can neither link the communicating parties nor decrypt their secret information exchanged [28, 29, 30]. Thus, hiding users' identities and the content communicated, the Tor network also prevents censorship.

However, compromising the Tor is an easy task, as the path selection algorithm in Tor prefers ORs with high bandwidth, i.e., lack randomness. Tor uses serial transmission and the same IP tunnel for multiple sessions, i.e., lack diversity. Moreover, the main goal of today's censorship is to prevent users from connecting to Tor by blocking the public ORs or jamming Tor traffic, i.e., affecting reliability and integrity. Typically, censorship can be done with a few major points in the network that filter, block, and jam IP traffic. As a countermeasure, Tor has introduced bridges, in the form of non-public ORs that are known to censored users only, and provide them with an entry into the Tor [32]. However, a censoring entity can collect and block these bridges as well. Thus, in fact, Tor does not protect against censoring entities themselves, which not only block and jam Tor traffic interrupting communication but also degrade the Tor performance. That indicates a need for additional randomness, diversity, coding and over-provisioning to defy modern censors.

To improve Tor performance and to randomize the tunnel (circuit) distribution, Tor multipathing has already been proposed, whereby

the bandwidth from different Tor circuits is aggregated to provide a high throughput [33, 34, 35, 36]. However, the studies on Tor focus on its performance by assuming loss-free networks and, thus, censorship by traffic jamming or circuit blocking remains an issue in real networks. The parallel transmission with coding and redundancy can protect from both jamming and eavesdropping. When the source data is combined with random information and sent over diverse paths, only the receivers are able to decode. Adversaries can only decode packets if they have received a sufficient number of data from multiple paths, which an eavesdropper might not be able to do. In the case of jamming, e.g., censorship, coding and path redundancy can protect the source data from errors and losses.

In this chapter, we investigate the security level of advanced network systems and the impact of parallel transmission, erasure coding, e.g., Linear Network Coding (LNC), random routing and over-provisioning on security. The data forwarding, here, is based on circuit switching or packet switching concept following either optimal or random routing presented in Chapter 2. A novel combinatorial approach enables the evaluation of security and reliability in case of eavesdropping and jamming attacks. In addition, we introduce *security degree* as a probability that an attacker can not recover sent data at the first try or disturb the communication. First, the physical layer attacks are studied. Here, the effectiveness of erasure coding is evaluated against balancing the demands of reliable transmission in the case of jamming and privacy in the case of eavesdropping. Finally, the censorship in the Tor network is considered as a type of jamming attacks. Here, we derive the probabilities of censorship success and deanonymization, and privacy degree. For that reason, a standard Tor network is extended to random routing, path and traffic parallelism, coding and over-provisioning concepts. All proposed analytical models take into account diverse security threats, which can occur simultaneously and in different combinations.

4.2 Supporting Publications

1. A. Engelmann and A. Jukan, "Balancing the demands of reliability and security with linear network coding in optical networks," 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016, pp. 1-7. doi: 10.1109/ICC.2016.7511590

2. A. Engelmann, S. Zhao and A. Jukan, "Improving Security in Optical Networks with Random Forwarding and Parallel Transmission," 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, 2015, pp. 1-6. doi: 10.1109/GLOCOM.2015.7417629

3. A. Engelmann and A. Jukan, "Defying Censorship with Multi-Circuit Tor and Linear Network Coding," 2019 12th CMI Conference on Cybersecurity and Privacy (CMI), Copenhagen, Denmark, 2019, pp. 1-6. doi: 10.1109/CMI48017.2019.8962147

4.3 Security Analysis of Parallel Transmission with Random Routing

The reference network architecture based on an optical network is shown in Fig. 4.1. The traffic is modeled as a binary sequence, decomposed into data blocks of the same length. The serial data flow of 16 data blocks is parallelized into multiple sub-flows through *Block Distribution*. We consider the circuit and packet switching, which differ in the way how packets are forwarded and routed in the network. In the case of circuit switching, the parallel routes are pre-reserved and constant during a certain transmission, while the data block distribution follows Round Robin. In the case of random packet switching, the data blocks are also parallelized but distributed randomly over as many parallel lanes as there are outgoing links

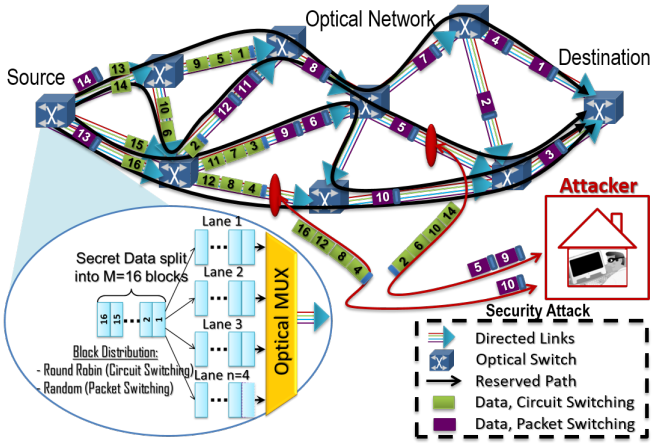


Figure 4.1: Wiretapping in optical networks with parallel transmission and random routing [8].

on the node, whereby the same forwarding operation is performed at every node in the network. Thus, each data block is forwarded randomly over any available output link towards the next hop and can utilize any out of all existing end-to-end paths.

Let us now assume that an eavesdropper (wiretapper) has access to multiple links in the network. In optical networks, the wiretapper can access all wavelength links inside any wiretap fiber link. In Fig. 4.1, there are two wiretap fiber links, whereby two out of four circuits (lightpaths) are compromised. The wiretapper, however, can only eavesdrop a half of the data sent. In contrast, in the case of random packet switching, the wiretapper would only receive a few packets, as they are randomly dispersed over the network.

For modeling and analysis, we use the models of network, switching and routing as presented in Sections 2.3.1 and 2.3.2. The notation used is shown in Sec. 2.7. The traffic is a binary sequence decomposed into M data blocks, which are then distributed over $k = n$ lanes to be sent over n pre-reserved circuits or any out of

$|\mathcal{E}_{out}(v)|$ output links in any network node v. In the case of circuit switching, the number of parallel sub-flows n corresponds to the number of required circuits in the network, i.e., each path carries M/n data blocks. In the case of packet switching, (optical) packets are sent over a randomly selected sequence of outgoing links, while any existing path can carry from 0 to M data blocks. Thus, to send data following the packet switching principle, at most M out of Ψ partly disjoint paths can be utilized, whereby each path carries at least one data block. Generally, each intermediate node v can forward incoming data blocks over a pre-configured set of $|\mathcal{E}_{out}(v)|$ outgoing links. Here, we do not consider wavelength conversion or buffering issues in optical networks and assume a loss-free network.

4.3.1 Threat Model

Let us assume that there is a collection $\mathbf{W}$ of sets $\mathfrak{W}$ of wiretap edges in the network $G(V, E)$, which are a subset of the edge set E, $\mathfrak{W} \in \mathbf{W}$ and $\mathfrak{W} \subset E$. The wiretapper can access only one $\varkappa^{th}$ set of collection $\mathbf{W}$, i.e., $\mathfrak{W}_{\varkappa}$, i.e., at most $\mathbf{w} = |\mathfrak{W}_{\varkappa}|$ edges in G at the same time. For an optical network with multiple wavelength links inside a fiber link (edge), we assume that all parallel wavelength links, which belong to a wiretap edge, can be compromised. Thus, there is a set $\mathfrak{W}'_{\varkappa}$ of all wiretap links, i.e., at most $|\mathfrak{W}'_{\varkappa}|$ links can be wiretapped simultaneously.

4.3.2 Analysis of Security Degree

Here, the set $\mathcal{M}$ contains all possible path subsets $\mathcal{M}_{\mathfrak{N}}(\alpha)$ utilized in a circuit or packet-switched network, where $1 \leq \alpha \leq a_{\mathcal{M}}$. We analyze next the security degree of any path combination $\mathcal{M}_k(\alpha)$ or $\mathcal{M}_{\Psi}(\alpha)$ in the circuit- or packet-switched networks, respectively. As, in the random packet-switched networks, all Ψ existing paths are utilized for transmission with a certain probability, the cardinality

of $\mathcal{M}$ is 1, i.e., $|\mathcal{M}| = a_{\mathcal{M}} = 1$. The security degree is defined as a probability that the secret data composed by M data blocks can not be recovered by a wiretapper when $\mathcal{L}(\alpha)$ out of M data blocks were eavesdropped from $\mathbf{w}$ wiretap edges. Then, the security degree is

$$\mathcal{D}(\mathcal{L}(\alpha)) = 1 - \left(\frac{1}{2^{\mathfrak{h}b} - 1}\right)^{M-\mathcal{L}(\alpha)} \cdot \frac{1}{(M - \mathcal{L}(\alpha))!}, \tag{4.1}$$

where the first multiplier describes the probability of successful guessing of $M - \mathcal{L}(\alpha) \geq 0$ missing data blocks at the first try. This probability generally depends on the number and size of the missing data blocks. The assumption is that each data block is split into $\mathfrak{h}$ symbols of b bits. Thus, the wiretapper needs to guess the missing blocks out of all $(2^{\mathfrak{h}b} - 1)$ possible bit patterns. The second multiplier describes the probability to find a correct order of all guessed $M - \mathcal{L}(\alpha)$ data blocks at the first try.

Without loss of generality, the amount of eavesdropped data blocks is a function of wiretap paths. Any (wavelength) path $\mathcal{P}_l$ utilized for parallel transmission, i.e., $\forall l, m : \mathcal{P}_{l_m}(\alpha) \in \mathcal{M}_{\mathfrak{N}}(\alpha)$, is wiretapped, when at least one link ρ_{lh} out of all $\mathfrak{y}_l$ links which compose this path is a wiretap link, i.e., $\rho_{lh} \in \mathfrak{W}'_{\varkappa}$. Any path can be wiretapped only once also if it consists of multiple wiretap links. That is because the wiretapping on different links of the same path will not increase the amount of eavesdropped data. When at most $\mathbf{w}$ edges are wiretapped, the number of compromised parallel paths over any $\mathfrak{N}$ available utilized paths is determined as follows

$$\mathcal{Z}(\mathfrak{N}, \alpha) = \sum_{m=1}^{\mathfrak{N}} \mathcal{Y}(\mathcal{P}_{l_m}(\alpha)), \tag{4.2}$$

where the logical function $\mathcal{Y}(\mathcal{P}_{l_m}(\alpha))$ determines whether the certain path $\mathcal{P}_{l_m}$ collected in set $\mathcal{M}_{\mathfrak{N}}(\alpha)$ contains at least one wiretap link:

$$\mathcal{Y}(\mathcal{P}_{l_m}(\alpha)) = \begin{cases} 1, & \text{if } \sum_{h=1}^{\mathfrak{y}_l} z(\rho_{lh}) \geq 1, \\ 0, & \text{else}, \end{cases} \tag{4.3}$$

where function $z(\rho_{lh})$ verifies if the link ρ_{lh} of the path $\mathcal{P}_l$ is wiretapped; and given as follows

$$z(\rho_{lh}) = \begin{cases} 1, & \text{if } \rho_{lh} \in \mathfrak{W}'_{\varkappa}, \\ 0, & \text{else.} \end{cases} \tag{4.4}$$

Next, we derive the number of the data blocks $\mathcal{L}(\alpha)$ eavesdropped over $\mathbf{w}$ wiretap edges and $\mathcal{Z}(\mathfrak{N}, \alpha)$ wiretap paths.

Random Circuit Switching

The circuit-switched network utilizes $n = k$ parallel pre-reserved paths collected in a certain set $\mathcal{M}_k(\alpha)$ to transmit all M data blocks of the same source data. Thus, the number of wiretap parallel paths is given by Eq. (4.2) as $\mathcal{Z}(k, \alpha)$, i.e., $\mathcal{Z}(k, \alpha)$ blocks can be collected by wiretapper every time unit, while $M/n = M/k$ blocks can be eavesdropped on any wiretap path $\mathcal{P}_{l_m}(\alpha)$, when $\rho_{lh} \in \mathfrak{W}'_{\varkappa}$ and $\rho_{lh} \in \mathcal{P}_l$. Thus, the total number of eavesdropped data blocks is

$$\mathcal{L}(\alpha) = \frac{M \cdot \mathcal{Z}(k, \alpha)}{k}. \tag{4.5}$$

Obviously, that the data transmission over one circuit $n = 1$ can result in eavesdropping of all M data blocks if the circuit is established using at least one wiretap link.

Random Packet Switching

In case of random packet switching over optical network, there are $\mathcal{Z}(\Psi, \alpha)$ (Eq. (4.2)) wiretap paths over all Ψ available paths, which contain at least one wiretap link $\rho_{lh} \in \mathcal{P}_l$, $1 \leq h \leq \mathfrak{y}_l$, i.e., $\rho_{lh} \in \mathfrak{W}'_{\varkappa}$. Here, all Ψ existing paths are collected in $\mathcal{M}_{\Psi}(\alpha)$ and can be utilized simultaneously. When an eavesdropper simultaneously accesses $\mathbf{w}$ wiretap edges and $\mathcal{Z}(\Psi, \alpha)$ wiretap paths, at most $\mathcal{Z}(\Psi, \alpha)$ data blocks can be eavesdropped on each time unit. That is possible

due to different end-to-end delays of randomly utilized paths and a random path selection for each data block.

Generally, each path $\mathcal{P}_l$ is described by the probability P_l as determined by Eqs. (2.8) and (2.13). Since, using random packet switching, a path is randomly selected for each out of M data block, the probability that at least one wiretap paths is randomly chosen and utilized any time unit during the transmission is determined as

$$P_{\mathcal{L}}(\alpha) = \sum_{l=1}^{\Psi} \mathcal{Y}(\mathcal{P}_{l_m}(\alpha)) P_l, \tag{4.6}$$

where $\mathcal{Y}(\mathcal{P}_{l_m}(\alpha))$ defined by Eq. (4.3) verifies if a certain path $\mathcal{P}_{l_m}(\alpha)$ is a compromised path. Thus, the amount of eavesdropped data blocks $\mathcal{L}(\alpha)$ sent over at most $\mathcal{Z}(\Psi, \alpha)$ wiretap paths is

$$\mathcal{L}(\alpha) = P_{\mathcal{L}}(\alpha) \cdot M. \tag{4.7}$$

4.4 Security Analysis of Parallel Transmission with Coding

This section uses the model of the circuit switching with optimal and random routing presented in Sections 2.3.1 and 2.3.2, the coding model from Sec. 2.4.1 and some notations from in Sec. 2.7.

Fig. 4.2 illustrates the reference network model, where traffic is modeled as a binary sequence, whereby any original data can be decomposed into M data blocks of the same length. Every data block is then distributed in a round robin fashion into k sub-flows at the source nodes. Moreover, Fig. 4.2 illustrates how LNC is performed at the end-system, i.e., at source and destination. The assumption that there is no coding in the middle of the network is valid here, as additional coding inside the network can only increase the security and reliability of the transmission system [92, 93]. The coding process is performed over a finite field F_{2^b}. Thus, each data block is decomposed into $\mathfrak{h}$ symbols, and each symbol has the same length of

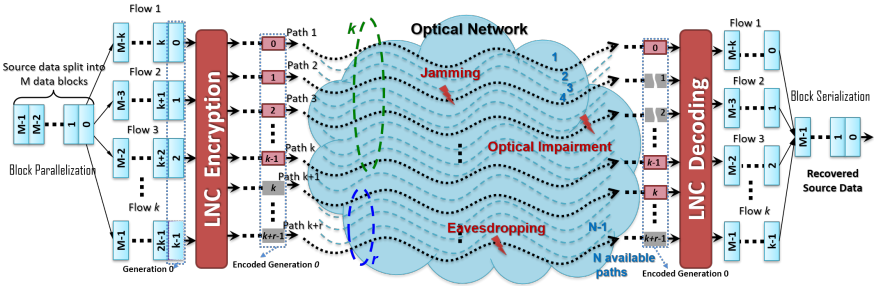


Figure 4.2: Security attacks in optical networks with the parallel transmission, coding and random routing [9].

b bits, i.e., the block length is $\hbar b$. Any k original symbols are encoded together into $n \geq k$ encoded symbols belonging to a certain generation. After encoding, the n coded parallel symbols correspond to the number of required lightpaths over the optical circuit-switched network. In other words, the source sends the encoded data over n out of all N available paths. When the number of encoded sub-flows n is larger than number of incoming sub-flows k, $n \geq k$, $r = n - k$ additional redundant symbols are sent over additional redundant paths. With redundancy, reliable transmission of one generation is straightforward when any k out of $n = k + r$ encoded symbols/data blocks from the same generation from any out of n paths are received, which implies that fully successful decoding at the receiver is possible. This feature provides protection against jamming attacks as well as any physical impairments when an attacker corrupt less than r data blocks from each generation. Also, eavesdropping attacks can be successfully defeated by coded parallel transmission because a wiretapper needs at least k encoded data blocks from the same coded generation distributed over a network as well as the knowledge about coding coefficients to recover the secret data.

4.4.1 Network Model

Here, the network provides F possible paths between source and destination, while at most N optical paths are available (i.e., non-blocked) for a circuit setup. For a successful data transfer, the choice of n, $n \leq N$, paths can be either optimal or random. We study both methods and evaluate the benefits of coding and path selection for security and reliability purposes. In contrast to the model in Sec. 2.3.1, we assume here that all F existing paths can be available with a certain probability $P_{\text{setup}} \equiv P_l^{\text{s}}$, where $1 \leq l \leq F$ and $P_1^{\text{s}} \leq ... \leq P_l^{\text{s}} \leq ... \leq P_F^{\text{s}}$. For parallel transmission, the source first selects n either optimal or random path out of F known paths to the certain destination and then tries to setup (reserve) these paths. Then, the set $\mathcal{M}$ of all possible path subsets $\mathcal{M}_{\mathfrak{N}}(\alpha)$ has a cardinality of $|\mathcal{M}| = a_{\mathcal{M}} = C_{F,\mathfrak{N}}$, where $1 \leq \alpha \leq a_{\mathcal{M}}$ and variable $\mathfrak{N}$ defines a number of available paths and can take any value, i.e., $0 \leq \mathfrak{N}$. Thus, the probability that the n certain paths, i.e., a certain set $\mathcal{M}_n(\alpha)$, is chosen and available for setup, can be derived from Eq. (2.4) as

$$P_F(\alpha, \mathfrak{N}) = \prod_{m=1}^{\mathfrak{N}} P_{l_m}^{\text{s}}(\alpha) \prod_{\substack{l=1, \\ \mathcal{P}_l \notin \mathcal{M}_n(\alpha)}}^{F} (1 - P_l^{\text{s}}). \tag{4.8}$$

The probability that there is at least one set of $\mathfrak{N}$ available parallel paths in the network can be defined as follows

$$\hat{P}(\mathfrak{N}, \mathcal{M}) = \sum_{\alpha=1}^{a_{\mathcal{M}}} P_F(\alpha, \mathfrak{N}), \tag{4.9}$$

where the α^{th} set from the collection $\mathcal{M}$ contains all $a_{\mathcal{M}}$ path combinations of $\mathfrak{N}$ available paths with related setup probabilities P_l^{s}. Thus, the probability to randomly select and setup a certain path combination $\mathcal{M}_n(\alpha)$ over all $a_{\mathcal{M}}$ possible path combinations can be derived from Eqs. (4.8) and (4.9) as follows

$$P_{\text{rnd}}(\alpha, n) = \frac{P_F(\alpha, n)}{\hat{P}(n, \mathcal{M})}. \tag{4.10}$$

In the case of $N < n$ available paths, parallel transmission is not possible, i.e., a transmission request will be blocked with probability:

$$P_B(n) = \sum_{\mathfrak{N}=0}^{n-1} \hat{P}(\mathfrak{N}, \mathcal{M}). \tag{4.11}$$

As a result, the probability to setup n optimal paths over $\mathfrak{N}$ available paths is a probability that a set $\mathcal{M}_{\mathfrak{N}}(\alpha)$ contains at least n available paths over $\mathfrak{N}$ considered paths, i.e., transmission request is not blocked. Thus, the probability of n optimal paths available for transmission is defined as follows

$$P_{\text{opt}}(\alpha, \mathfrak{N}) = \frac{P_F(\alpha, \mathfrak{N})}{1 - P_B(n)}, \tag{4.12}$$

where $\mathfrak{N}$ can vary as $n \leq \mathfrak{N} \leq F$.

4.4.2 Threat Model

Let us now extend the threat model from Sec. 4.3.1 assuming that there are two collections of the edges, i.e., $\mathbf{W}^E$ and $\mathbf{W}^J$. These two collections contain sets $\mathfrak{W}^E_{\varkappa}$ and $\mathfrak{W}^J_{\varkappa}$ of wiretap edges in the network $G(V, E)$, on which an attacker can perform eavesdropping or jamming attacks, respectively. Moreover, we assume that the wiretap edges are static and do not change in the course of our analysis, i.e., they were selected by a wiretapper only once. The attacker can access only one $\varkappa^{th}$ set from collection $\mathbf{W}^E$ or $\mathbf{W}^J$, i.e., $\mathfrak{W}^E_{\varkappa}$ or $\mathfrak{W}^J_{\varkappa}$. However, both types of attacks can be performed simultaneously on different or the same edges. Thus, an attacker can attack at most $\hat{\mathbf{w}} = \mathbf{w}^E + \mathbf{w}^J - \mathbf{w}^{EJ}$ edges in $G(V, E)$ at the same time, where $\mathbf{w}^E = |\mathfrak{W}^E_{\varkappa_E}|$, $\mathbf{w}^J = |\mathfrak{W}^J_{\varkappa_J}|$ and $\mathbf{w}^{EJ}$ is a number of eavesdropped and jammed edges, which simultaneously appear in $\mathfrak{W}^J_{\varkappa_J}$ and $\mathfrak{W}^E_{\varkappa_E}$. Moreover, we consider the worst-case scenario, where an attacker is able to access all wavelengths of a certain fiber link, which significantly increases the number of compromised links and wavelength paths in optical networks.

4.4.3 Analysis of Security Degree

To protect against jamming attacks, we propose to send over r additional paths r additional redundant sub-flows of coded data blocks. Thus, an attacker can recover or corrupt the secret data when the number of attacked parallel paths in a certain path set $\mathcal{M}_{\mathfrak{N}}(\alpha)$ is $\mathcal{Z}(n,\alpha) \geq k$ or $\mathcal{Z}(n,\alpha) > r$, respectively. For the eavesdropping attacks, moreover, the security degree can be analyzed with Eq. (4.1) which is a function of the eavesdropped data blocks $\mathcal{L}(\alpha)$ and their size $(\mathfrak{h} \cdot b)$. To transmit all M data blocks of the secret data, the source utilizes $n \geq k$ parallel pre-reserved paths, while each path carries at most M/k data blocks. Thus, $\mathcal{L}(\alpha)$ data blocks can be collected or corrupted by a wiretapper on $\mathcal{Z}(n,\alpha)$ wiretap paths. As a result, the total number of eavesdropped data blocks is determined by Eq. (4.5). Since the number of wiretap paths can vary as $0 \leq \mathcal{Z}(n,\alpha) \leq n$, i.e., $\mathcal{L}(\alpha) \geq M$, Eq. (4.1) needs to be modified as

$$\mathcal{D}(\mathcal{L}(\alpha)) = 1 - \left(\tfrac{1}{2^{\mathfrak{h}\cdot b}-1}\right)^{min\{M-\mathcal{L}(\alpha),0\}} \cdot \frac{1}{(min\{M-\mathcal{L}(\alpha),0\})!}. \quad (4.13)$$

The number of wiretap paths $\mathcal{Z}(n,\alpha)$ utilized for transmission and collected in a path set $\mathcal{M}_n(\alpha)$ can be calculated with Eq. (4.2). For dynamic scenarios, however, the number of wiretap paths in any path set $\mathcal{M}_n(\alpha)$ depends on the path selection method, as different paths can be available for random paths selection or optimization with a certain setup probability only. Thus, there is a need to derive a mean value of compromised paths over all possible path combinations $\mathcal{M}_n(\alpha)$ utilized for random or optimal routing out of all F existing paths between a certain source-destination pair.

Optimized Parallel Transmission

The mean number of wiretap paths utilized for the optimized coded parallel transmission can be derived considering the number of wiretap paths in any possible path combination $\mathcal{M}_n(\alpha)$, i.e., $\mathcal{Z}(n,\alpha)$

(Eq. (4.2)), and the probability that optimized parallel transmission is possible if any $\mathfrak{N}$ paths out of F paths are available, i.e., $P_{\text{opt}}(\alpha, \mathfrak{N})$ (Eq. (4.12)). Thus, the mean number of wiretap paths utilized for optimized parallel transmission is defined as follows

$$\bar{\mathcal{Z}}_{\text{opt}} = \sum_{\mathfrak{N}=n}^{F} \sum_{\alpha=1}^{C_{F,\mathfrak{N}}} P_{\text{opt}}(\alpha, \mathfrak{N}) \cdot \mathcal{Z}(n, \alpha), \tag{4.14}$$

where $C_{F,\mathfrak{N}}$ defines the number of all possible combinations of $\mathfrak{N}$ available paths over all F existing paths.

Random Circuit Switching

In case of random path selection, any n available paths out of F existing paths collected in a certain set $\mathcal{M}_n(\alpha)$ can be selected with equal probability and utilized for transmission as described by probability $P_{\text{rnd}}(\alpha, n)$. However, the mean value of wiretap paths over all possible path combinations collected in $\mathcal{M}$ can be derived considering any path combination $\mathcal{M}_n(\alpha)$ and the number of wiretap paths in it $\mathcal{Z}(n, \alpha)$. Thus, the mean number of utilized wiretap paths is

$$\bar{\mathcal{Z}}_{\text{rnd}} = \sum_{\alpha=1}^{C_{F,n}} P_{\text{rnd}}(\alpha, n) \cdot \mathcal{Z}(n, \alpha), \tag{4.15}$$

where the probability of n certain parallel paths selected randomly $P_{\text{rnd}}(\alpha, n)$ and the number of wiretap paths out of n selected $\mathcal{Z}(n, \alpha)$ are determined by Eqs. (4.9), and (4.2), respectively.

4.4.4 Catastrophic Security Threat

To analyze the impact of path selection on security, let us consider the worst-case scenario, where there is only one wiretap edge in the network, and several paths are allocated to go over it. That leads to what we refer to as *catastrophic security threat*, which results in security degree $\mathcal{D}(\mathcal{L}(\alpha)) = 0$ or in jamming success. As a measure

of catastrophic security threat, we define two probabilities related to eavesdropping and jamming attacks. For eavesdropping, we evaluate the probability that a wiretapper can receive from one wiretap edge at least $\nu = k$ LNC encoded data blocks from each generation and encode the whole secret data. In the case of jamming attacks, the wiretapper is able to corrupt at least $\nu = r + 1$ data blocks from each generation transmitted and, thus, to prevent successful decoding at the destination. In other words, the secret data can experience a catastrophic security threat when ν utilized parallel paths are established over the same wiretap edge. The probability of catastrophic security threat $\Gamma_{\mathrm{opt}}(\nu, n)$ in case of optimized coded parallel transmission over n parallel paths is determined as follows

$$\Gamma_{\mathrm{opt}}(\nu, n) = \sum_{\mathfrak{N}=n}^{F} \sum_{\alpha=1}^{C_{F,\mathfrak{N}}} P_{\mathrm{opt}}(\alpha, \mathfrak{N}) \cdot \mathcal{X}(\alpha, \nu, n), \tag{4.16}$$

where function $X(\alpha, \nu, n)$ is a logical function, which verifies valid path combinations $\mathcal{M}_n(\alpha)$ with at least ν wiretap paths:

$$\mathcal{X}(\alpha, \nu, n) = \begin{cases} 1, & \text{if } \mathcal{Z}(n, \alpha) \geq \nu, \\ 0, & \text{else.} \end{cases} \tag{4.17}$$

Finally, the probability of catastrophic security threat $\Gamma_{\mathrm{rnd}}(\nu, n)$ in the case of random routed coded parallel transmission over n randomly selected paths can be derived as follows

$$\Gamma_{\mathrm{rnd}}(\nu, n) = \sum_{\alpha=1}^{C_{F,n}} P_{\mathrm{rnd}}(\alpha, n) \cdot \mathcal{X}(\alpha, \nu, n). \tag{4.18}$$

4.5 Analysis of Anonymity, Privacy and Censorship Success

4.5.1 Threat Model

In this section, we introduce different types of attacks on the Tor network and, thus, different types of adversaries, their approaches and goals. Section 4.5.5 shows utilized notations.

Deanonymization Attack

The goal of a deanonymization attack is to reveal the identity of either client (initiator of anonymous communication) or server (receiver of requests for anonymous communication) or both. For that reason, an adversary usually compromises Onion Routers (OR) or configures its own ORs to be selected by a client as an entry or exit node. The adversaries, who access an entry OR, can observe the connection between the client and their entry, i.e., deanonymize the client. The adversaries, who access the exit OR, on the other hand, can observe traffic between the exit OR and the server, i.e., deanonymize the server. If both the entry OR and exit OR are compromised and cooperate, they can use traffic analysis to link the client to the server, which we refer to as a *total deanonymization*.

Censorship Attack

The fundamental tools of censorship are filtering and blocking of websites, IP addresses, etc., network disconnection or jamming. Thus, all these interventions result in connection or traffic loss. Tor network of today can not prevent that the onion traffic, the onion routers and the directory authorities are blocked, for example by the so-called *"Great Firewall of China"* [32]. To enable entry to the anonymous network despite such blocking, the Tor network has introduced bridges, i.e., unlisted onion routers used as entry nodes.

These bridges are distributed to censored users via out-of-band mechanisms. However, a censor can recognize, collect and finally block the bridges. Some censors, moreover, can reside in ORs or access nodes and links of the underlying network to perform jamming of Tor traffic. Any circuit blocking and traffic loss will require complex recovery mechanisms at the destination and traffic retransmission. We propose to address the latter with Linear Network Coding (LNC).

Privacy Attack

Tor is a fundamentally privacy assuring network. Privacy, however, can still be compromised by eavesdropping and traffic analysis. The assumption, here, is that any single or cooperative eavesdropper can reside in ORs or the underlying physical networks, compromising network links and nodes. The privacy attack has two steps. First, an adversary needs access to Tor circuits, and then, in the second step, tries to decrypt the eavesdropped data. For parallel transmission, the adversary needs to collect traffic distributed over multiple circuits, which potentially increases the privacy of the Tor network implemented based on principles of the advanced network systems.

4.5.2 Design and Modeling of Multi-Circuit Anonymous Network with Coding

To counter the threats described above, we make use of the diversity of circuits and onion routers traversed while deploying random circuit switching and coding. Three Tor architectures are studied:

Traditional Tor Architecture

The basic Tor network is shown in Fig. 4.3 and involves Onion Proxies (OP) and OR. OP presents a client (Alice), who is an initiator of the anonymous communication and uses information about existing ORs to set up a circuit to a server (Bob). Some ORs in

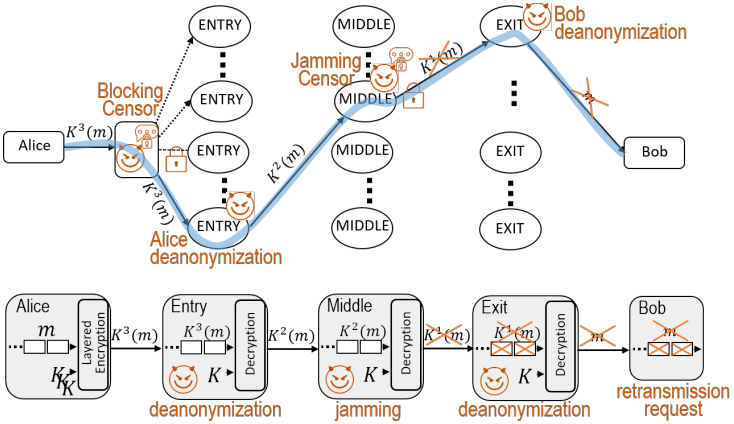


Figure 4.3: Tor with serial transmission [10].

the Tor network are compromised, or there is a censor, who filters traffic and blocks connections to some entry ORs, i.e., *blocking censor*. Alice randomly chooses three ORs (entry, middle and exit) and constructs a circuit. In this example, the selected entry OR is unknown to the blocking censor, while all selected ORs are nevertheless compromised: entry deanonymizes Alice identity, the middle OR is accessed by *jamming censor*, who disturbs Tor traffic, and the exit deanonymizes Bob identity. When the same adversary compromises entry and exit nodes, both Alice and Bob can be recognized as communication endpoints. That results in the mentioned *total deanonymization*. In general, the traffic m is split into fixed-sized (512-byte) units called *cells*. The cells are encrypted in layers as $K^3(m)$ with three different keys K^i of three selected ORs, which can remove only one encryption layer applying the same key K^i as Alice. Thus, ORs decrypt and forward the cells to the next OR or Bob, outside of the Tor network. When the middle node jams cells and the exit node does not detect erroneous cells, the exit decrypts the jammed cells and forwards them to Bob, who can not recover the data and requests cell retransmission.

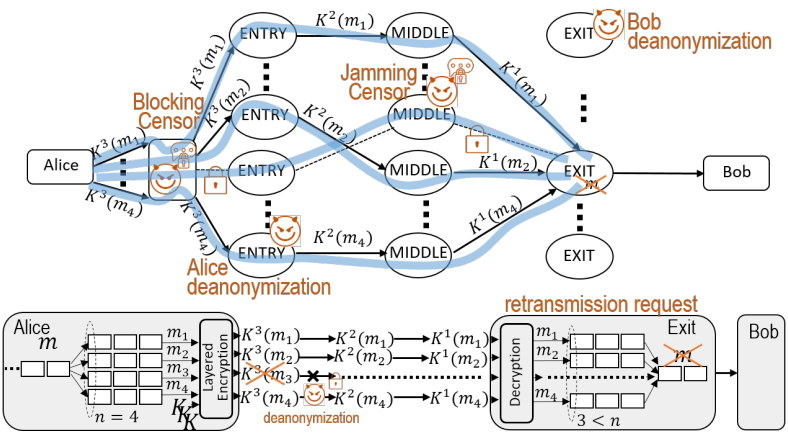


Figure 4.4: Tor with multiple parallel circuits [10].

Multiple-Circuit Tor (mTor) Architecture

Since the selection algorithm of ORs in Tor favors routers that have higher bandwidths, it leads to poor randomness during circuit setup increasing the circuit predictability. In contrast, the traffic splitting over multiple circuits improves load balancing and throughput, whereby ORs and circuits can be selected randomly and independently of other performance parameters. Fig. 4.4 shows Tor implementation with multiple circuits. Alice splits incoming traffic m into cells of fixed size and builds $n = 4$ traffic sub-flows, i.e., m_1, m_2, m_3, m_4. Alice randomly selects $2n + 1 = 9$ ORs and setups $n = 4$ disjoint circuits, which only share an exit router. Any parallel traffic sub-flow m_i is then encrypted in layers with keys of ORs from related circuit i and sent over this circuit toward the exit node. The exit node collects all sub-flows, decrypts and serializes them into original data m before sending it to Bob. In Fig. 4.4, the first and second circuits, which transport encrypted sub-flows m_1 and m_2, respectively, are honest. The third circuit, however, can not deliver sub-flow m_3 since the entry node is blocked by a blocking censor,

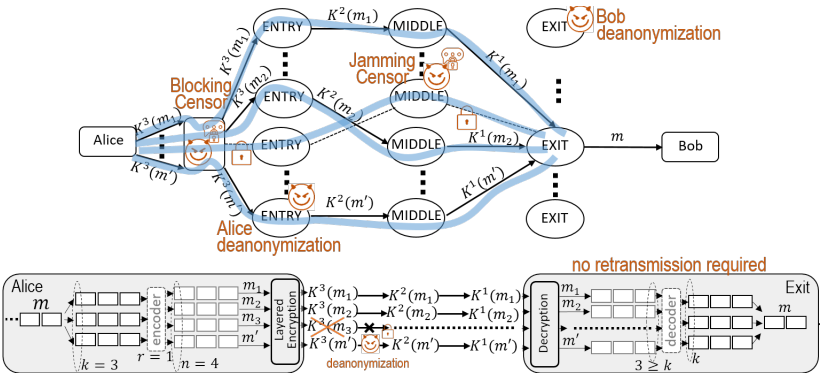


Figure 4.5: Tor with multiple parallel circuits and erasure coding [10].

which prevents Alice from entry into the Tor network. The m_3 will not be delivered also when the entry node was not blocked by blocking censor, because of compromised middle node, where a jamming censor disturbs Tor cells. As a result, the exit node is unable to recover original data m due to missing sub-flow m_3 and has to request a retransmission of m_3. Moreover, Alice selected a compromised entry OR for the fourth circuit and, thus, will be deanonymized.

Multiple-Circuit Tor with Coding (cTor)

Here, LNC can be used as an erasure code to protect traffic against losses. Fig. 4.5 illustrates a possible application of LNC in the Tor network, whereby Tor is implemented with multiple circuits and parallelized traffic. Similar to mTor from Fig. 4.4, the ORs are selected randomly and independently of other performance parameters. Alice randomly selects $2n + 1 = 9$ ORs and setups $n = 4$ parallel disjoint circuits, which only share an exit router. Also, here, one entry OR will deanonymize Alice, and the third circuit will be blocked by the blocking censor or disturbed by jamming censor, due to the known to the censor entry node or compromised middle node, respectively. Generally, any (n, k) erasure code encodes k units of original data

into n units of coded data and tolerate a loss of up to $r = n - k$ data units from the same generation, whereby the original and encoded data units have the same predefined size. Thus, Alice splits incoming traffic m into cells of fixed size and parallelizes them to build $k = 3$ cell flows, e.g., m_1, m_2, m_3. The LNC encoder takes one cell from each cell flow and encodes these $k = 3$ cells together to generate $r = 1$ redundant cell, i.e., call flow m'. After encoding, $n = k + r$ coded cells leave the encoder building $n = 4$ parallel cell flows, i.e., m_1, m_2, m_3, m'. Each coded cell flow is finally encrypted in layers to be sent over a certain circuit to exit OR. Due to censorship, the third circuit with a cell flow m_3 is interrupted resulting in a loss of m_3. Circuit failures and cell losses, however, do not affect the throughput of cTor as long as at least $k = 3$ circuits can deliver k coded cells from each generation to the exit node. In Fig. 4.5, the third circuit fails to deliver cells, whereby other circuits (at least $k = 3$) provide cells for generation recovery. During the decoding process, any $k = 3$, e.g., m_1, m_2, m', out of $n = 4$ cells from each generation can recover the original data, whereby any out of $r = 1$ redundant cells m' from a certain generation can replace any lost cells. Finally, the recovered cells can be serialized into the original traffic flow m and sent to Bob. To reduce latency and required buffer to compensate out-of-order effect, we propose to start the decoding process as soon as k coded cells from the same generation are decrypted and reached the decoder. The remaining cells from the slower circuits can be ignored without any processing and buffering.

4.5.3 Routing model

We assume that there are M honest and M' compromised low-bandwidth ORs, M_b secret bridges and M'_b bridges known to the blocking censor, and M_e honest and M'_e compromised high-bandwidth exit ORs. Any uncensored user randomly selects $2n$ ORs out of existing $M + M'$ ORs and one exit OR out of $M_e + M'_e$, while any censored user randomly selects n entry nodes out of $M_b + M'_b$ bridges, n middle ORs out of $M + M'$ existing ORs and one exit OR. We assume that M'' out of M' low-bandwidth ORs and M''_e out of M'_e high-bandwidth exit ORs are controlled by the same adversary to perform the total deanonymization.

After selecting $2n + 1$ ORs, n disjoint circuits are established between the client and the exit node and one circuit between the exit node and the server. Generally, the circuits are setup randomly over a randomly selected set of ORs. We assume that the network provides N_α honest and N'_α compromised available parallel circuits between client and server over any set α of the randomly selected entry, middle and exit ORs, where $1 \leq \alpha \leq n$. Moreover, any set of ORs α provides the same number of honest and compromised circuits $N_1 = ... = N_n = N$ and $N'_1 = ... = N'_n = N'$, respectively. Then, one circuit among $N + N'$ circuits is randomly selected for circuit setup over certain ORs. Thus, there are in total $\mathcal{N} = \sum_{\alpha=1}^{n}(N_\alpha + N'_\alpha) = n(N + N')$ available parallel circuits to establish anonymous communication over n circuits. Moreover, N'' out of N' compromised circuits can cooperate with each other and $M'' + M''_e$ compromised cooperative ORs.

4.5.4 Security Analysis

Next, we analyze the probability of successful deanonymization, successful censorship by blocking or jamming censor, as well as the privacy degree of Tor, mTor, and cTor.

Deanonymization Success

We can distinguish between three deanonymization attacks: 1) Client deanonymization, whereby a compromised entry OR reveals client identity to an adversary; 2) Server deanonymization, whereby a compromised exit OR reveals the server identity, its services, and the content exchanged; 3) Total deanonymization when both the entry and the exit nodes are compromised and cooperate to link the client to the server doing traffic analysis.

The probability of a successful client deanonymization can be determined as the probability that at least one randomly selected entry OR is compromised. Since all entry ORs are chosen randomly with the same probability out of all $M+M'$ available ORs, the probability of client deanonymization is given as follows

$$P_c = \frac{1}{\binom{M+M'}{n}} \sum_{i=1}^{min\{n,M'\}} \binom{M}{n-i}\binom{M'}{i}, \tag{4.19}$$

whereby we consider all $\binom{M+M'}{n}$ possible combinations of ORs as entry nodes against combinations with at least one compromised entry OR, i.e., i compromised ORs, where $1 \leq i \leq min\{n, M'\}$. Since the client randomly selects only one exit OR out of $M_e + M'_e$ ORs, the probability of successful server deanonymization is

$$P_s = \frac{M'_e}{M_e+M'_e}, \tag{4.20}$$

where one out of M'_e compromised ORs is selected against all $M_e + M'_e$ candidates ORs. Finally, the probability that both client and server can be linked is a probability that the client randomly selects

at least one compromised entry OR out of M'' ORs and one compromised exit OR out of M''_e ORs. Based on Eqs. (4.19) and (4.20), the probability of total deanonymization is determined as follows

$$P_{c-s} = \frac{M''_e}{(M_e+M'_e)\binom{M+M'}{n}} \sum_{i=1}^{min\{n,M''\}} \binom{M+M'-M''}{n-i} \binom{M''}{i}, \qquad (4.21)$$

where M'' and M''_e is a number of compromised cooperative ORs.

Censorship Success

Censorship is successful if anonymous communication is successfully disrupted. Thus, we classify censorship as 1) blocking of entry ORs (bridges), i.e., circuit blocking, and 2) jamming of Tor traffic at network nodes and links or by compromised ORs, i.e., the receiver is not able to recover the information sent.

Blocking of entry ORs, i.e., bridges, results in the blocking of the whole Tor circuit and the loss of the entire cell flow sent over the circuit. In the case of anonymous communication over Tor and mTor, the communication between client and server is disturbed if at least one bridge utilized is known to a censor and blocked. The client randomly selects n bridges, where $n \geq 1$, out of $M_b + M'_b$ available bridges, whereby the probability of choosing a bridge that is known to the censor and blocked, i.e., the probability of a censorship success, can be calculated similarly to Eq. (4.19) and derived as

$$P_{bb} = \frac{1}{\binom{M_b+M'_b}{n}} \sum_{i=1}^{min\{n,M'_b\}} \binom{M_b}{n-i} \binom{M'_b}{i}, \qquad (4.22)$$

where i, $1 \leq i \leq min\{n, M'_b\}$, out of n bridges are blocked.

When LNC is applied, i.e., cTor, the censorship is only successful if $i > r$, $r+1 \leq i \leq min\{n, M'_b\}$ and $M'_b > r$, out of n used bridges are known to the censor and blocked. Then, the destination receives less than k, i.e., $n - i < k$, cells from each generation and can not

recover the original information by decoding. Here, $n-i$ out of M_b unknown bridges and i out of M_b' known bridges are selected as n out of $M_b + M_b'$ entry nodes. In contrast, cTor communication can not be disrupted by bridge blocking, if $M_b' \leq r$. Similar to Eqs. (4.19) and (4.22), the probability of successful censorship is given as

$$P_{bb}^{\mathrm{LNC}} = \frac{\sum_{i=r+1}^{min\{n,M_b'\}} \binom{M_b}{n-i}\binom{M_b'}{i}}{\binom{M_b+M_b'}{n}}, \; M_b' > r. \tag{4.23}$$

When censorship is performed as jamming of cell flow on a certain circuit to reduce throughput or to interrupt anonymous communication, we consider an adversary, who can reside in M' ORs and network nodes and links compromising in total $n \cdot N'$ circuits. Thus, the anonymous communication over Tor and mTor will be disrupted if the client selected at least one compromised OR or one compromised circuit. Considering the probability that all $(2n+1)$ randomly selected ORs and all n circuits are honest, the probability of successful censorship with jamming is determined as follows

$$P_{\mathrm{jam}} = 1 - \frac{\binom{M}{2n}}{\binom{M+M'}{2n}} \cdot \frac{M_e}{M_e + M_e'} \cdot \left[\frac{N}{N+N'}\right]^n. \tag{4.24}$$

Applying LNC, the anonymous communication over cTor can not be damaged as long as at least k out of n circuits and their ORs are honest and deliver cells to the decoder. To determine the probability of successful censorship by jamming, we first define the probability $P_{\mathrm{OR}}(f, n, M, M')$ that f out of n established circuits from client to exit node are compromised. The exit OR has to be honest. Otherwise, the censorship success, i.e., probability of communication disruption, will be defined by Eq. (4.20). Any circuit will be compromised when at least one OR on it is compromised. Thus, the probability of f compromised circuits can be determined as follows

$$P_{\mathrm{OR}}(f, n, M, M') = \binom{n}{f} \sum_{s=f}^{min\{2f,M'\}} \frac{\delta\psi(f,s,n)M!M'!(M+M'-2n)!}{(M-2n+s)!(M'-s)!(M+M')!}, \tag{4.25}$$

where the binomial coefficient defines all possible combinations of the compromised and honest circuits, and the sum considers a probability to have s, $f \leq s \leq min\{2f, M'\}$ compromised ORs on f compromised circuits. For $s = f$, each out of f compromised circuits contains one compromised OR. Thereby, $\delta = 1$ if $2n - s \leq M$ and $\delta = 0$ else, to ensure, that s compromised ORs can exist on f compromised circuits. The function $\psi(f, s, n)$ defines a number of all possible allocations of the compromised and honest ORs over n circuits and is calculated as follows

$$\psi(f,s,n) = \begin{cases} 1, & \text{if } s = 2f \text{ or } f = 0, \\ 2^f, & \text{if } s = f, \\ f \cdot 2^{(2f-s)}, & \text{if } (f < s < 2f) \& (s \neq \frac{3f}{2}), \\ & \text{or } (s = \frac{3f}{2}) \& (f < n), \\ s \cdot 2^{(2f-s)}, & \text{if } (s = \frac{3f}{2}) \& (f = n). \end{cases} \tag{4.26}$$

The circuits established through a selected set of the compromised and honest ORs can also be compromised by a censor, who resides in the underlying network. The probability $P_{\text{CH}}(f, n, N, N')$ that f out of n established circuits are censored due to some compromised nodes and links in the network is determined as follows

$$P_{\text{CH}}(f,n,N,N') = \binom{n}{f} \left[\tfrac{N}{N+N'}\right]^{n-f} \left[\tfrac{N'}{N+N'}\right]^{f}. \tag{4.27}$$

Thus, the probability of successful censorship by jamming of more than r circuits of cTor can be calculated as follows

$$\begin{aligned} P_{\text{jam}}^{\text{LNC}} =& 1 - \tfrac{M_e}{M_e+M'_e} \sum_{f=0}^{r} \sum_{i=0}^{min\{f,M'\}} P_{\text{OR}}(i,n,M,M') \cdot \\ & \cdot \sum_{j=f-i}^{f} P_{\text{CH}}(j,n,N,N') \gamma(f,i,j,n), \end{aligned} \tag{4.28}$$

where $\gamma(f, i, j, n)$ defines the probability that i compromised circuits due to compromised ORs and j compromised circuits due to network

adversary are selected by a client resulting in f compromised circuits. Thereby, some circuits can be compromised by both adversaries, who reside in ORs, and other nodes and links of the underlying network.

$$\gamma(f,i,j,n) = \begin{cases} \frac{\binom{n-j}{f-j}\binom{j}{i-(f-j)}}{\binom{n}{i}}, & \text{if } j \geq i, \\ \frac{\binom{n-i}{f-i}\binom{i}{j-(f-i)}}{\binom{n}{j}}, & \text{if } j < i, \end{cases} \tag{4.29}$$

where the first case ($j \geq i$) is for the scenario where there are more compromised circuits due to network adversaries and, in the second case ($j < i$), there are more compromised circuits due to compromised ORs. Here, condition $f \leq j + i$ is valid.

Privacy Degree

We consider privacy degree as a probability that any single or cooperative adversaries who reside in onion routers or the underlying network can not recover secret information transmitted in cells despite all secret keys and coding coefficients are known. To compromise secret information of Tor and mTor transported over n, $n \geq 1$, circuits without LNC, the adversary needs an access to all n circuits with probability $P_{\text{CH}}(n, n, N + N' - N'', N'')$ or their ORs with probability $P_{\text{OR}}(n, n, M + M' - M'', M'')$ defined by Eqs. (4.27) and (4.25), respectively, to collect cells distributed over these n circuits. After collecting any $f < n$ cell flows from f compromised circuits, an adversary needs to guess $(n - f)L$ cells of $512 \cdot 8$ bits out of all $2^{512 \cdot 8(n-f)L}$ possible bit sequences to recover the anonymously sent information, where L is a number of cells sent over one circuit. We assume that there is no privacy if the exit OR is compromised since the adversary can access all n cell flows. Based on a probability of successful collection of f cell flows and guessing all missing cells at

the first try, the privacy degree of Tor and mTor is determined as

$$\Delta P = 1 - \sum_{f=0}^{min\{n,M''+N''\}} \Delta p(f) \sum_{i=0}^{min\{f,M''\}} P_{\mathrm{OR}}(f,n,M''',M'') \cdot \\ \cdot \frac{M_e}{M_e+M'_e} \sum_{j=f-i}^{min\{f,N''\}} P_{\mathrm{CH}}(n,n,N''',N'')\gamma(f,i,j,n), \tag{4.30}$$

where $M''' = M+M'-M''$, $N''' = N+N'-N''$ and $\Delta p(f) = \frac{1}{2^{512\cdot 8(n-f)L}}$ is a probability of successful guessing of $(n-f)L$ missing cells. Using cTor, the cooperative adversary needs to access only $k = n - r$ out of n circuits to reveal entire secret information. If the adversary accesses $f < k$ circuits, the $(k-f)L$ cells need to be guessed. The privacy degree of cTor is based on a probability of successful collection of f cell flows and guessing all $(k-f)L$ missing cells at the first try, and can be calculated as follows

$$\Delta P^{\mathrm{LNC}} = 1 - \sum_{f=0}^{min\{n,M''+N''\}} \Delta p^{\mathrm{LNC}}(f) \sum_{i=0}^{min\{f,M''\}} P_{\mathrm{OR}}(f,n,M''',M'') \cdot \\ \cdot \frac{M_e}{M_e+M'_e} \sum_{j=f-i}^{min\{f,N''\}} P_{\mathrm{CH}}(n,n,N''',N'')\gamma(f,i,j,n), \tag{4.31}$$

where $\Delta p^{\mathrm{LNC}}(f) = \frac{1}{2^{512\cdot 8\cdot max\{(k-f),0\}L}}$ is a probability of successful guessing of missing cells and $\gamma(f,i,j,n)$ is determined by Eq. (4.29).

4.5.5 List of Symbols

m	secret message of Alica;
k	number of uncensored circuits required to recover a secret message;
r	number of redundant Tor circuits;
n	total number of Tor circuits required, i.e., $n = k + r$;
M	number of honest ORs with a low bandwidth;
M'	number of compromised ORs with a low bandwidth;
M''	number of compromised ORs out of M' ORs controlled by the same adversary;
M_b	number of bridges unknown to blocking censor;
M'_b	number of compromised bridges known to censor;
M_e	number of honest exit ORs with a high bandwidth;
M'_e	number of compromised exit ORs of a high bandwidth;
M''_e	number of compromised exit ORs out of M'_e ORs controlled by the same adversary;
N	number of possible honest Tor circuits;
N'	number of possible compromised Tor circuits;

4.6 Performance Evaluation

This section numerically evaluates the overall security degree provided by parallel transmission, coding and random routing; and studies their impact on anonymity, privacy and censorship success.

4.6.1 Overall Security Degree

First, the physical layer threats are assumed according to Sections 4.3 and 4.4. All data blocks have the same size of 8 bits, and the secret data is composed of $M = 100$ data blocks. The transmission rate of each optical link is defined as 10 Gbps. Thus, the resulting *time unit (tu)* amounts $0.8ns$. The network is assumed to be lossless, i.e., the data always reach the destination. The theoretical results presented were validated by event-driven simulations. The normalized confidence intervals show 95% confidence and small values, and thus could not be presented with error bars. For the simulations and analysis, we used the NSFnet network presented in Fig. 2.7. The link directions were defined as follows $\{0-1, 0-2, 0-7, 1-2, 1-3, 2-5, 3-10, 3-4, 4-5, 4-6, 5-9, 5-13, 6-7, 7-8, 8-11, 8-12, 9-8, 10-11, 10-12, 11-13, 12-13\}$, where nodes 0 and 13 are the source and destination nodes, respectively.

Here, we refer to circuit switching as PAR-EXP and to packet switching as PAR-RND. To maximally increase the security degree, $n = 4$ or $n = 8$ maximally edge-disjoint paths are selected for PAR-EXP. We assumed that four edges, $\mathfrak{W} = \{3-10, 5-13, 8-12, 4-6\}$, can be wiretapped simultaneously, or at different time points, and in different combinations. Thus, we consider a scenario where only one, two, three or all four edges are wiretapped at the same time. That results in $\mathbf{w} \in 1, 2, 3, 4$ and collection $\mathbf{W}$, which consists of 4, 6, 4 and 1 elements, respectively. Based on these assumptions, we

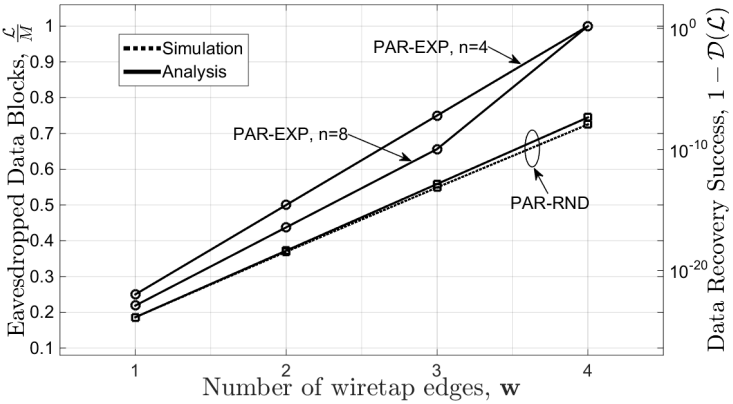


Figure 4.6: Amount of eavesdropped data and security degree [8].

evaluate the amount of eavesdropped data blocks and the security degree, in cases of PAR-EXP and PAR-RND transmission calculated with Eqs. (4.5), (4.7) and (4.1), where $\mathcal{L}(\alpha) \equiv \mathcal{L}$.

The results in Fig. 4.6 present the mean number of eavesdropped data blocks and the security degree, which were calculated over all possible edge combinations from $\mathbf{W}$. Here, the number of eavesdropped frames $\mathcal{L}$ increases with an increasing number of wiretap edges $\mathbf{w}$ and reach 100% in case of PAR-EXP transmission and 4 wiretap edges. Generally, the mean number of eavesdropped data in the case of PAR-RND was always less than in the case of PAR-EXP and showed values of around 20% and 73% for $\mathbf{w} = 1$ and $\mathbf{w} = 4$, respectively. Fig. 4.6 shows the probability of successful recovery of secret data at the first try, i.e., $(1 - \mathcal{D}(\mathcal{L}))$, based on the results for $\mathcal{L}$. The probability of successful recovery of the secret data increases with an increasing number of wiretap edges, e.g., from 10^{-25} to 10^{-7} for PAR-RND and from 10^{-23} to 1 for PAR-EXP if the number of wiretap edges was $\mathbf{w} = 1$ and $\mathbf{w} = 4$, respectively. In the case of PAR-EXP, the security degree can be improved for a small number of wiretap edges by increasing the number of parallel edge-

disjoint paths, i.e., n. However, a wiretapper can eavesdrop all M data blocks regardless of n, i.e., $n = 8$ and $n = 4$, if there are $\mathbf{w} = 4$ wiretap edges at the same time.

Next, we numerically investigate the impact of erasure coding (LNC) on resulting security in optical circuit-switched networks (PAR-EXP) with optimal and random routing, i.e., PAR-EXP-OPT and P-EXP-RND simplified as OPT and RND, respectively. The LNC process was implemented over a finite field F_{2^8} with a symbol size of 8 bits and a block size of 10 symbols. For the simulations and analysis, we use the NSFnet topology with links directed as shown in Fig. 2.7. The total number of parallel wavelength paths was set to $F = 18$, whereby the number of wavelength paths inside each fiber paths defined in Tab. 2.1 can be summarized as $\{4, 2, 6, 2, 1, 1, 0, 1, 1, 0, 0\}$. All wavelength paths have the end-to-end delay of the related fiber paths, i.e., determined by the number of hops. The parallel paths have different probabilities of being available at the moment of transmission request. For the sake of the analysis, we assumed that these probabilities decrease with increasing path length, i.e., $\vec{P}_{\text{setup}} = \{0.8, 0.79, 0.78, 0.77, 0.70, 0.69, 0.60, 0.59, 0.58, 0.57, 0.56, 0.55, 0.54, 0.53, 0.50, 0.45, 0.40, 0.35\}$. In the case of RND, n out of $N \leq F$ available parallel paths are selected randomly. In the case of OPT, the data flows are sent in parallel over n optimized (shortest) parallel paths. To evaluate the probability of a catastrophic security threat, we assume that the wiretap edge is the edge with the largest capacity utilized to establish the shortest (optimal) paths between source and destination, i.e., the edge between node 2 and node 5.

Fig. 4.7 shows the probability that an attacker can recover or corrupt the whole secret data by accessing those edge, where both y-axes on the left and right sides show the results for a successful jamming and eavesdropping attack, respectively. The eavesdropping attack is successful when the generation size k was small, i.e., $k = 4$. For OPT, the probability of catastrophic eavesdrop-

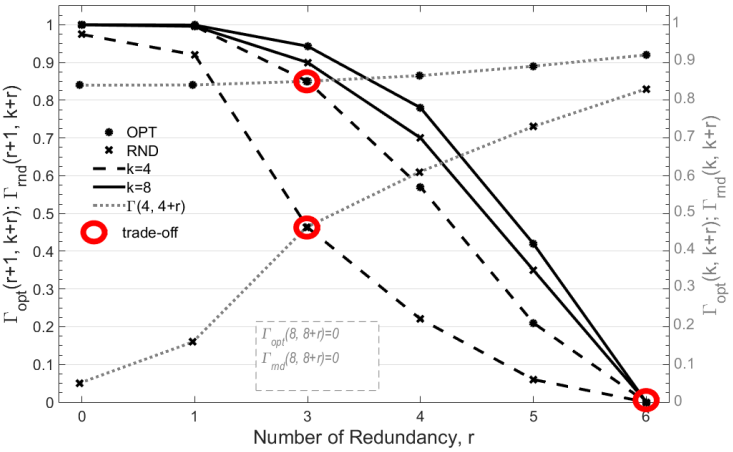


Figure 4.7: Probability of catastrophic security threat [9].

ping slightly increased with increasing redundancy r from 84% to 92% for $r = 0$ and $r = 6$, respectively. For RND, the probability of catastrophic eavesdropping increased from 5% to 83% with an increasing r. In contrast, the attacker gained no knowledge about secret data regardless of path selection method and amount of redundancy, when the generation size was set to $k = 8$, i.e., $\Gamma_{\text{opt}}(8, 8 + r) = \Gamma_{\text{rnd}}(8, 8 + r) = 0$. On the other hand, jamming is successful if an attacker corrupted more than r data blocks from the same generation, and the receiver is not able to decode. Thus, an attacker prevents data recovery at the destination if no or only one redundant data block is generated to protect each generation of $k = 8$ data blocks. The increase in redundant data blocks and a decrease in generation size k decreases the probability of catastrophic jamming. RND with $k = 4$ shows the best performance against jamming attacks. With $r = 6$ redundant data blocks, the probability of catastrophic jamming can be reduced up to 0% for all transmission methods and all values of k. In contrast to the probability of catastrophic jamming, the probability of catastrophic eavesdropping

increases with an increasing amount of redundancy. However, there is a trade-off for both methods, RND and OPT, when an attacker simultaneously performs eavesdropping and jamming on the same edge. Generally, the risk of the catastrophic security threat can be balanced out for both attacks and each transmission method. For instance, when an initial generation size and redundancy were set to $k = 4$ and $r = 3$ data blocks, respectively, the probabilities of catastrophic jamming and eavesdropping threat showed equal values of 46% and 84% for P-RND and P-OPT, respectively. For $k = 8$, $r = 6$ redundant data blocks are required to eliminate the risk of successful eavesdropping and jamming attacks.

4.6.2 Anonymity, Privacy and Censorship Success

Now, we show the analytical results validated by Monte-Carlo simulations for the Tor network extended to parallel transmission, random routing, coding, and path and coding redundancy. We analyze a generic network topology, whereby any client can select between $M + M' \geq 25$ ORs, $M_e + M'_e \geq 12$ exit ORs, $M_b + M'_b \geq 25$ bridges and $N + N' \geq 12$ possible paths over any selected set of ORs to setup one circuit. Additionally, there exist 175 low bandwidth ORs, which can be utilized only in the case of parallel transmission.

Fig. 4.8 shows the probability of client deanonymization as a function of the amount of compromised ORs M', whereby the number of honest ORs M was set to 25 ORs. Here, mTor and cTor can additionally utilize the low bandwidth ORs. The deanonymization probability increases with an increasing number of compromised ORs and parallel paths n. Generally, mTor and cTor implementation can provide the highest client anonymity, while Tor outperforms mTor and cTor implemented with $n = 6$ and $n = 9$ circuits.

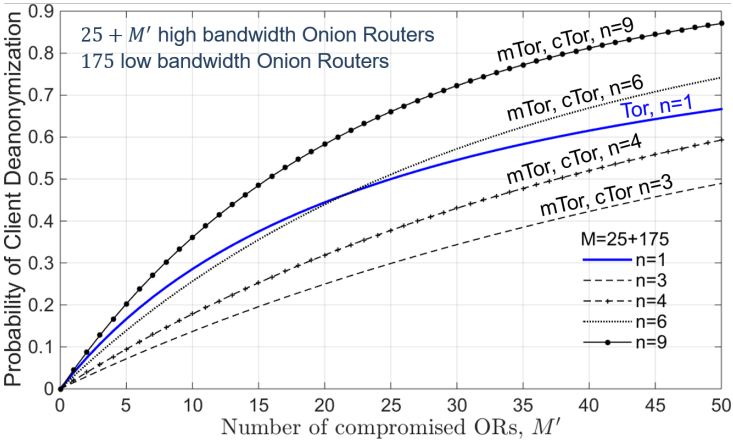


Figure 4.8: Probability of client deanonymization.

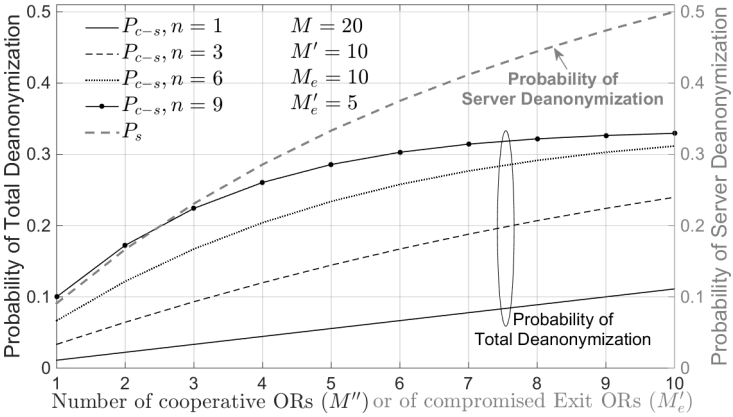


Figure 4.9: Probability of server deanonymization P_s (gray) and total deanonymization P_{c-s} (black) [10].

In contrast, the probability of server deanonymization P_s depends on the number of compromised exit nodes M'_e only as shown in Fig. 4.9 with gray color and reaches 50% when $M_e = M'_e = 10$. Fig. 4.9 also illustrates the probability of total deanonymization P_{c-s}, i.e., both client and server are linked to each other by compromised cooperative ORs. Here, we assumed $M_e = M'_e = M''_e = 5$ exit ORs and $M+M' = 20+10$ other ORs and considered communication over different number of parallel circuits n. The probability of total deanonymization slightly increases with increasing M'' and rises with a number of parallel circuits n, while it does not exceed 33%. Thus, Tor provides the highest overall anonymity as the probability of selecting one compromised circuit is lower than the probability of selecting at least one out of $n > 1$ compromised circuits.

Next, we consider both censorship attacks by blocking censor and jamming censor. Fig. 4.10 shows the probability of successful communication interruption due to blocking of entry bridges, whereby we assumed that $M_b = 25$ bridges are unknown to the censor. The probability of circuit blocking increases with the increasing number of utilized circuits n for Tor and mTor. The mTor over $n = 8, 10$ circuits will be blocked with probability 100% when the censor knows more than 15 bridges. In contrast, cTor shows much better performance, which depends on the amount of utilized coding redundancy r, e.g., cTor always outperforms mTor and Tor if $M'_b \leq 25$. The communication over $n = 10$ and $n = 5$ circuits incl. $r = 4$ and $r = 2$ redundant circuits (40% redundancy) shows the lowest probability of communication interruption as long as $M'_b < 15$ and $15 \leq M'_b < 25$, respectively. In case of a jamming censor, cTor and mTor can use 175 low bandwidth ORs. Here, cTor outperforms both Tor and mTor as presented in Fig. 4.11, where transmission over $n = 10$ and $n = 5$ circuits incl. $r = 4$ and $r = 2$ redundant circuits, respectively, shows the best results. However, P_{jam} and P_{jam}^{LNC} increases upto100% and up to 82%, respectively, with increasing M' and n. In cTor,

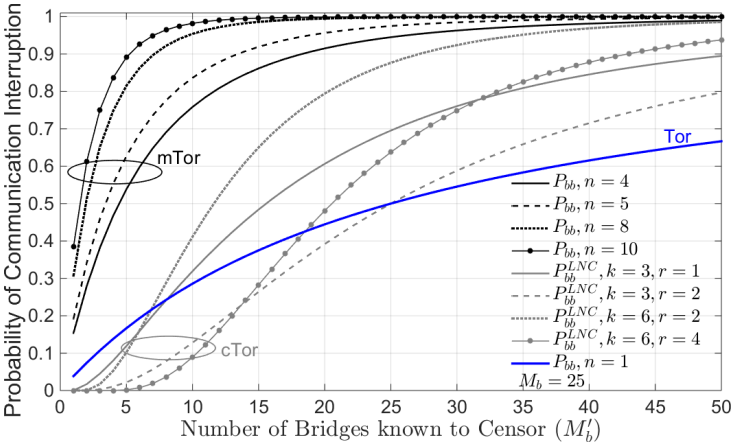


Figure 4.10: Probability of successful communication interruption due to blocking of bridges [10].

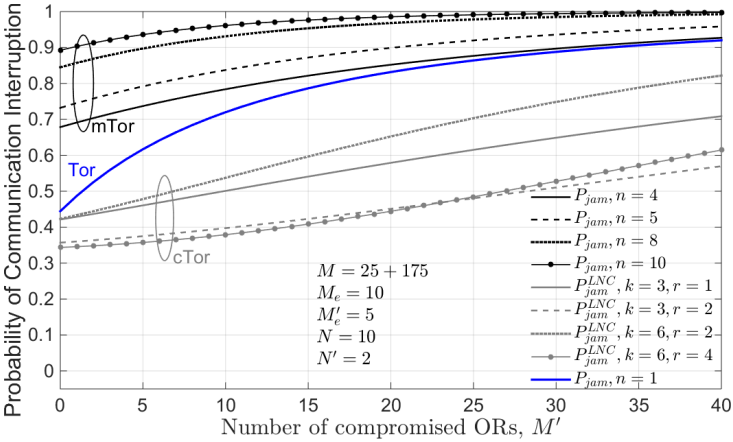


Figure 4.11: Probability of successful communication interruption due to jamming of Tor traffic.

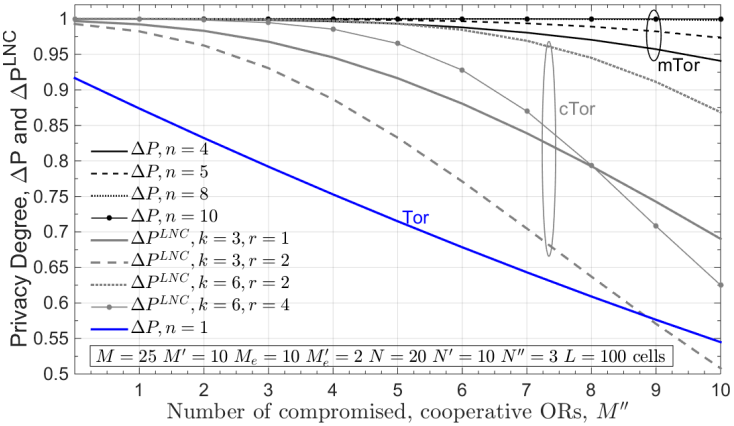


Figure 4.12: Privacy Degree [10].

increasing r reduces the probability of communication interruption. The standard Tor outperforms mTor only.

Finally, Fig. 4.12 shows a privacy degree for anonymous communication with Tor, mTor, and cTor as a function of cooperative eavesdroppers M'', who reside in ORs or the network and access $N'' = 3$ circuits over any selected set of ORs. The privacy degree ΔP of Tor and mTor increases with an increasing number of the utilized circuits n. As the redundant circuits in case of cTor enclose additional cells to the eavesdropper, privacy degree ΔP^{LNC} of cTor is slightly reduced as compared to mTor, but still outperforms Tor.

4.7 Summary

In this section, the impact of parallel transmission with random routing and coding on anonymity, privacy, integrity and censorship was investigated in the presence of eavesdropping and jamming adversary. First, the effect of random routing on privacy in packet-switched and circuit-switched networks was analyzed and compared. The results showed that random packet switching can provide a high security degree when also multiple physical links are compromised. On the other hand, the random circuit switching showed better security performance in the networks with a single wiretap edge and a large number of available edge-disjoint paths.

Then, the impact of random routing and erasure coding on privacy and reliability was studied in circuit-switched networks. Here, we analyzed the coding capability to balance reliability and privacy of parallel transmission when eavesdropping and jamming are performed simultaneously on different physical links. The results confirmed that coding and parallel transmission over randomly selected paths significantly increases security level against both eavesdropping and jamming as compared to standard multipath routing. Moreover, the conventional multipathing can lead to a *catastrophic security threat*, whereby an attacker can eavesdrop or corrupt a whole secret data by accessing only one edge in a network.

Finally, novel Tor implementations, i.e., (multi-circuit) mTor, and (coded) cTor, were introduced, modeled and analyzed regarding anonymity, privacy, integrity and reliability, whereby adversaries could reside in ORs and the underlying physical network and eavesdrop, jam or block Tor traffic. The analytical and simulation results proved that cTor with the parallel transmission, eraser coding, redundancy and a full-random selection of ORs and circuits provides sufficient anonymity and significantly increases privacy, integrity and reliability in the case of censorship as compared to traditional Tor.

5

Engineering and Benchmarking Anonymous Communication Systems

5.1 Introduction

The anonymity of communication is defined by unlinkability and unobservability of communicating end-points and ensured through layered encryption. In the most known anonymous network, i.e., The Onion Routing (Tor) [31], the layered encryption is implemented with the well-known Advanced Encryption Standard (AES) based on the Rijndael algorithm [94]. It presents a practically secure symmetric key encryption [95] in combination with the Diffie-Hellman key exchange algorithm [96]. Today's Tor networks are evolving at a fast pace to address the vulnerabilities caused by network performance bottlenecks, including throughput and latency. However, the path selection in Tor is particularly critical, as it favors computationally powerful onion routers sending Tor traffic over one circuit only. Thus, randomness and diversity of the Tor network are limited by design resulting in poor anonymity, integrity, and reliability.

This thesis postulates that optical networks can help anonymous networks to further evolve and improve upon performance. To improve randomness and diversity, optical networks can choose among hundreds of wavelengths over a single fiber. Akin to multipath

routing, optical networks also provide communication channel parallelism intrinsically [2, 97, 98]. Since Tor is based on a concept of a connection-oriented communication channel, *a circuit*, Wavelength Division Multiplexing (WDM) or elastic optical channels can be used. The only feature that the optical network could not perform today is layered encryption all-optically. Even though today's encryption standards can operate at optical line rates [99] and can be efficiently used on point-to-point links, i.e., without encryption in the middle of the network, optical data in an anonymous optical network would have to be converted into the electronic domain and processed electronically. Such electronic processing in optical nodes is typically characterized by significant data storage, processing latency as well as energy dissipation. Hence, for the entire concept of all-optical encryption to work in the middle of the network, en-/decryption with an optical stream cipher is a necessary feature.

The open question is how feasible is an implementation of the optical stream cipher. Implementing today's AES in optics would require an extensive number of logic gates, high complexity of synchronization of optical components to ensure requirements on the algorithm's complexity and timing. A new approach to all-optical encryption similar to symmetric key cryptography in electronics [95, 100] is essential. Current approaches in optical cryptography are not suitable to this end, as they randomize parameters of the optical laser, such as phase and amplitude of signal or to insert a pseudorandom noise into the optical signal [101, 102, 103, 104] which cannot be used for layered encryption. Even if AES implemented in FPGA were able to solve issues of scale, memory, latency, and energy, AES hardware implementations are known to be vulnerable against differential power analysis and fault injection attacks [105].

In this chapter, we investigate the feasibility of a novel method for all-optical layered encryption with an all-optical stream cipher (oSC) based on the optical XOR (oXOR) and optical LFSR (oLFSR)

components. Our main idea is to generate a long optical keystream at an optical line rate based on the shorter electronic secret at comparably lower bit rates. We design a cascaded cryptosystem based on the proposal in [37], i.e., a cryptographically secure subsystem is cascaded with a cryptographically insecure subsystem. As any cascade provides the overall security of its cryptographically most strong subsystem, we propose to use a standard cryptographically secure, but low-rate pseudo-random number generator as defined by ENISA [106], NIST [107] and ISO/IEC 18031 [108] with insecure oLFSR operating at line rate. It was shown that an optical cascaded cryptosystem requires new security evaluation techniques due to the required bit rate adaptation. For that reason, novel security benchmarks suitable for optical cascaded cryptosystems were developed based on the information theory and allow us the engineering of an anonymous optical network with a configurable security level.

5.2 Supporting Publications

1. A. Engelmann and A. Jukan, "Toward All-Optical Layered Encryption: A Feasibility Analysis of Optical Stream Cipher," in IEEE Transactions on Information Forensics and Security, vol. 14, no. 10, pp. 2689-2704, Oct. 2019. doi: 10.1109/TIFS.2019.2904793

2. A. Engelmann and A. Jukan, "Practical privacy in WDM networks with all-optical layered encryption," 2017 IEEE International Conference on Communications (ICC), Paris, 2017, pp. 1-6. doi: 10.1109/ICC.2017.7996813

3. A. Engelmann and A. Jukan, "Optical Onion Routing," 2017 International Conference on Computing, Networking and Communications (ICNC), Santa Clara, CA, 2017, pp. 323-328. doi: 10.1109/ICCNC.2017.7876148

5.3 Background

Let us motivate the idea of all-optical encryption with an example of a traditional Tor-over-IP-over-optical network and its development towards all-optical layered encryption. In Fig. 5.1a, three onion routers (A_1, A_2 and A_3) are chosen. Here, the path between Alice and Bob consists of three concatenated IP tunnels (circuits), i.e., between Alice and onion router A_1, A_1 and A_2 and A_2 and A_3. During the path setup, three different session keys are distributed to all onion routers selected, and three onion encryption layers are wrapped around the secret message m using AES. The onion routers A_1, A_2 and A_3 disguise IP addresses of source and destination by replacing them, thus ensure unlinkability. Each onion router removes one encryption layer by applying the same key as Alice. As it is commonly the case, the transmission between routers is implemented as point-to-point fiber connections while all data processing, including layered encryption, is performed electronically, via IP packet header processing and AES encryption.

When the layered encryption is implemented all-optically, the electronic data processing is only required at edge routers A_1 and A_2 (Fig. 5.1b). Here, A_1 and A_2 disguise IP addresses of source and destination, while all-optical layered encryption is performed in optical anonymization nodes (oA1, oA2, oA3, and oA4). In contrast to traditional Tor, a number of optical anonymization nodes can be increased without a significant impact on latency or network throughput. Furthermore, wavelength routing on optical fibers can be highly randomized, which is today possible and straightforward, by using a private, confidential control plane and network management information [109]. The control plane of optical networks can be separated from the data plane in the form of known Software Defined Networks (SDN) [110, 111], making a circuit set up a standard procedure.

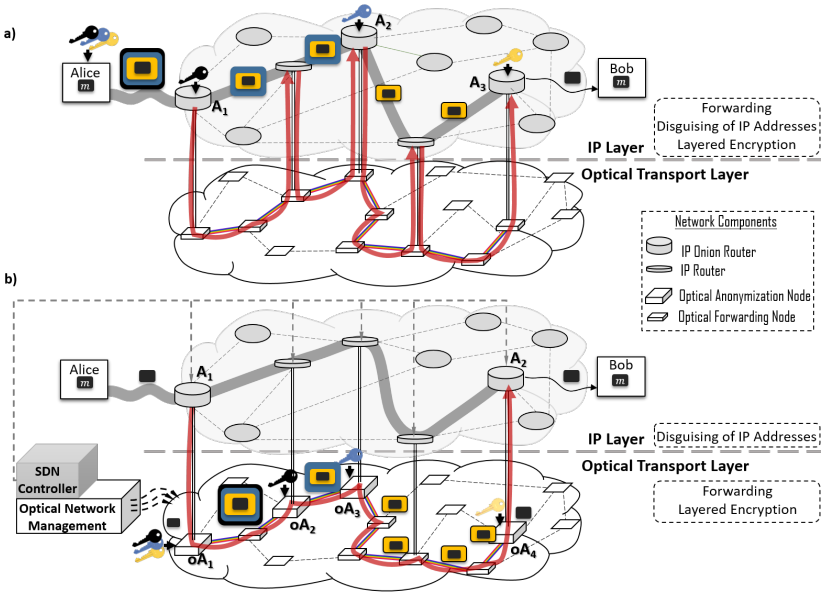


Figure 5.1: Onion Routing in IP-over-optical networks [11]: a) electronic IP packet routing and AES encryption b) optical circuit switching and all-optical layered encryption.

In cryptography, the encryption key defines the power of an encryption algorithm [112]. The perfect encryption key for a stream cipher is a "one-time pad" [113, 114] presented by a stream of random bits with a high grade of entropy, whereby this keystream needs to be as long as the secret message and utilized only once for encryption. That perfectly secure encryption (with a key at least as long as message) defined by Shannon in [113] was relaxed in [115]; the *perfect security* was considered in a computationally-bounded world. In [115], semantic security was introduced, for the case that, using a known ciphertext, an adversary can reveal a secret message out of all known secret messages with probability only negligibly higher than without knowing a ciphertext. The semantically secure encryp-

tion does not require an encryption key as long as a secret message if there is an algorithm to securely expand any short key. The entropic security used for analyses in this thesis was introduced in [116] and extended in [117] representing a weakened version of semantic security by the assumption that any secret message has enough uncertainty (high entropy) from the adversary's point of view.

Modern symmetric key cryptography, i.e., block ciphers such as AES and stream ciphers, utilizes a short secret bitstream, that can be divided into a secret key and an initialization vector, to perform two operations: 1) generation of long keystream based on that secret input stream, and 2) bit-by-bit encryption (XOR concatenation) of a stream of plaintext with a keystream generated. Stream ciphers utilize the secret key and initialization vector with a fixed length for the state initialization of the keystream generator. In general, a secure and efficient keystream generator designed in electronics presents a combination of Linear Feedback Shift Registers (LFSRs), Feedback with Carry Shift Registers (FCSRs), Non-Linear Feedback Shift Registers (NLFSRs), T-functions, lookup tables and different logic gates as shown in ECRYPT Stream Cipher Project (eSTREAM) [118] and, thus, is designed akin to cryptographically secure electronic pseudo-random number generator (pRNG) as defined by [107, 108]. In late 2004, eSTREAM selected and announced seven efficient and secure stream ciphers, which are either software-oriented such as C-128, Rabbit, Salsa 20/12 and SOSEMANUK or hardware-oriented such as Grain v1, MICKEY, and Trivium [119].

Whereas the bit-by-bit encryption from these stream ciphers can be easily transferred to optical stream cipher using an oXOR gate, implementation of an optical key generator (oKG) is rather challenging for two reasons: technological immaturity of optical components and system complexity due to many optical gates required.

To design a key generator similar to AES or eSTREAM, we need to either have optical components that are technologically immature today, e.g., large optical buffers, or a large number of optical logic gates, making the all-optical implementation rather impractical. For instance, AES, HC-128, and SOSEMANUK are implemented with S-boxes, which would require optical buffer [120, 121]. The Rabbit would require multiple optical adders, i.e., numerous XOR, AND and OR optical gates [122, 123], thus significantly increase implementation complexity; Salsa 20/12 requires both optical storage and optical adders [124]. Grain, Trivium and MICKEY are based on NLFSRs, which are not available in photonics today, etc. [125, 126, 127, 128]. For these reasons, we need a new design of an optical key generator with as few state-of-the-art optical logical gates as possible while being able to generate long optical keys.

A possible practical solution to generate a long optical key from a short secret bitstream is to deploy the previously mentioned optical LFSR (oLFSR) at line rate [129, 130, 131]. Indeed, there are implementations of oLFSR with a register length of up to 32 bits at rates of up to 250 Gbit/s. Clocking signal in an oLFSR is implemented by an RZ laser that generates an optical signal at a line rate. This technique allows oLFSRs not only to extend a short input stream (the so-called seed of n bits) into a particular output sequence with a maximal length of $2^n - 1$ bits, endlessly repeated, but also to adapt a bit rate of an optical key to the operating optical speed of oLFSR. Since LFSRs are well-known as cryptographically insecure due to intrinsic linearity, the generation of a cryptographically strong and long key requires some sort of de-linearization. In electronics, for instance, outputs of multiple LFSRs are combined through different nonlinear functions, i.e., logic circuits based on AND, OR, and XOR gates or look-up tables presenting de-facto a pseudo RNG (pRNG) [112]. However, an all-optical implementation of pRNG at a line rate is a challenge for current optical technology, as discussed above.

To address the issues of linearity, key length and high bit rate, an approach known as a cascade of random systems can be used akin to a proposal in [37], which has proven security such that: any cascade is at least as difficult to cryptographically break as its most-difficult-to-break component. A cascade was first defined in [132] as a chain of block or stream ciphers. The plaintext of any cascade is the plaintext of the first cipher, and the ciphertext of any i^{th} cipher is the plaintext of the $(i+1)^{th}$ cipher in the cascade. As a result, the ciphertext of the cascade is the ciphertext of the last cipher. In contrast to [132], which proved the security of cascade based on the block ciphers only assuming that an attacker cannot exploit information about the plaintext statistics, [37] generalized a cascade as a chain of any random systems and proved its security considering known-plaintext, chosen-plaintext, chosen-ciphertext attacks when an attacker knows the plaintext statistics. It was shown that cascade based on stream ciphers is at least as difficult to break as the most-difficult-to-break component cipher. As a result, a keystream sequence that is generated by cascaded ciphers with independent keys (seeds) is at least as difficult to predict as the most-difficult-to-predict keystream sequence. Moreover, [37] considered cascading of a small set of keystream generators, each relying on a different design principle, rather than to employ one large keystream generator. The cascaded key generator can then only fail if all applied design principles fail simultaneously. The design of some practical ciphers, e.g., Grain and Trivium [119] which use cascade connections of NLFSRs as their main building blocks, has been motivated by the idea that if one of the subsystems in a cascade is strong then the whole cascade is strong. Thus, we design, next, an oKG as a cascade based on a cryptographically secure subsystem, and for that we choose a low bit rate cryptographically secure electronic pRNG, granting the cryptographic security; and high bit rate cryptographically insecure oLFSRs, granting the operation at optical line speed.

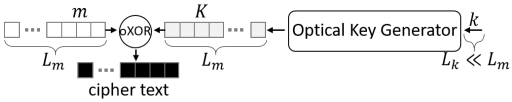


Figure 5.2: Encryption with optical stream cipher (oSC) [11].

5.4 Engineering an Optical Stream Cipher

To realize an optical stream cipher (oSC), the function of encryption and decryption can be performed by a simple oXOR function between optical data m, and the optical key K as illustrated in Fig. 5.2. We use the term *optical data* to refer to digital electronic data modulated by Amplitude Shift Keying (ASK) or on-off keying. ASK encodes electronic data by turning on or off the amplitude of light. Each binary symbol ("1" or "0") is represented by the presence or the absence of light. It includes non-return to zero (NRZ), return to zero (RZ), duo-binary formats and other variations of the RZ.

The stream cipher encrypts optical data m of length L_m by an encryption key of the same size $L_K = L_m$. The decryption of received encrypted data is then implemented with the same key K and oXOR operation. Since the length of optical data transmitted is in the range of hundreds of Gb/s and the optical encryption key K needs to be at least of the same length, it is necessary to generate a long optical key at optical line rate just before en-/decryption in any anonymization node along the optical circuit. In this way, there is no need to buffer the keys in optical storage or its additional transmission. Therefore, an oKG must operate at optical line rates to be able to generate a long optical key from a short, secure sequence k.

5.4.1 Optical Key Generation

For optical key generation, we assume state-of-the-art optical components: oXOR and oLFSR operating at the optical line rate C. Besides, we assume a cryptographically secure electronic pRNG usually

at a much lower bit rate $C_R << C$. pRNG can be based on any key generation algorithm of any stream cipher proposed by eSTREAM and generates a nonlinear bit sequence with a high grade of entropy.

Without loss of generality, any LFSR of length n which is based on a primitive irreducible polynomial of degree n can extend the so-called *seed* of n bits (a short input start sequence generated by pRNG) into an infinite long output sequence with an entropy of this seed. That is possible as an LFSR of length n can generate a unique bit sequence of maximal length $2^n - 1$ bits based on a certain seed and endlessly repeat these $2^n - 1$ bits. In our cascaded oKG, a secure subsystem, i.e., pRNG, can expand secure control sequence k of length L_k into a longer secure sequence M of length L_M at low transmission rate C_R; and the insecure subsystem based on oLFSR can expand a sequence M from pRNG into a very long optical key K simultaneously adapting its bit rate to the line rate C.

It is well-known that due to linear properties of LFSRs, an attacker is able to break any LFSR initialized by n bits with time complexity $O(n^2)$ and memory complexity $O(2n)$ applying the Berlekamp-Massey algorithm on $2n$ output bits generated by the LFSR. Considering this weakness, one could increase cryptographic security by bounding the length of output sequence from any oLFSR by $2n - 1$ bits. This solution, however, is challenging because of the large size of optical data and the long key required, i.e., $2n - 1 << L_K$; here, $L_K \to \infty$ requires either an infinitely long oLFSR register, i.e., $n \to \infty$, or an infinite number of parallel oLFSRs, $P \to \infty$, of short length, $n << \infty$. On the other hand, we assume that an attacker already knows oLFSR length n and its generator polynomial. That makes the reduction of oLFSR output to $2n - 1$ bits obsolete because an attacker needs to guess the input sequence, i.e., seed, to reproduce the output sequence. Instead of this reduction, we need to make sure that oLFSR does not generate the same bit sequences α of length $L_\alpha = 2^n - 1$ successively as it will significantly reduce

the quality of optical keystream as well as the level of diffusion and confusion. Let us denote a bit pattern of optical stream m, which consists of N finite optical sub-streams s_i, $1 \leq i \leq N$, of length L_s, as $m = [s_1, s_2, ..., s_N]$, where s_i and s_{i+1} can have the same or different bit patterns. The length of optical stream m can be then determined as $L_m = N \cdot L_s$. We refer to a key as *poor*, if, for instance, $K = [k_1, k_2, ..., k_j, ...] \equiv [k_1, k_1, ..., k_1, ...] = [\alpha, \alpha, ..., \alpha]$. To this end, the length L_α of key sequence α generated by a certain oLFSR must be $L_\alpha << 2^n - 1 << L_m$. To generate a long optical key with a high level of entropy and without sequence repeat, we hence need new methods for all-optical oLFSR *de-linearization*. All notations utilized in this chapter are summarized in Sec. 5.8.

Basic oKG (oKG-A)

Fig. 5.3a presents an oKG which consists of only one pRNG and one oLFSR. The secret parameter k initializes pRNG-a, which generates nonlinear bits $M = [\tilde{z}, \tilde{n}]$ at low bit rate C_{R_a}, where bit pattern $\tilde{z}$ and $\tilde{n}$ have length z and n, respectively. The first n out of L_M bits are used as a seed to initialize oLFSR, while further z bits are utilized for nonlinear shifting of oLFSR, i.e., $L_M = n + z$. Some of the bits from the optical shift register are XOR concatenated, as it is typically the case in LFSR. The resulting bits b are forwarded to an additional oXOR gate for concatenation with nonlinear $\tilde{z}$-bits from pRNG-a. Then, nonlinear bits $\tilde{b} = \tilde{z} \oplus b$ are inserted back to the shift register at the last position. As a result, bits in the register are shifted so that the bit on the first position leaves the register as a key bit of the optical encryption key K.

For an effective oLFSR de-linearization, pRNG-a needs to generate at least one bit before oLFSR generates α_a, $L_{\alpha_a} << 2^n - 1$ optical key bits, i.e., $\frac{L_{\alpha_a}}{C} \geq \frac{1}{C_{R_a}}$. Thus, the required bit rate of pRNG-a is

$$C_{R_a} \geq \frac{C}{L_{\alpha_a}}. \tag{5.1}$$

That results in z additional cryptographically strong bits from pRNG-a, and z nonlinear oLFSR shifting operations during the entire key generation process. The number of $\tilde{z}$-bits can be determined as

$$z = \frac{L_m}{L_{\alpha_a}}, \tag{5.2}$$

which increase the entropy of the generated optical key K.

The oKG-A has to be implemented considering two requirements, i.e., $C_{R_a} << C$ and $L_{\alpha_a} << 2^n - 1$. However, since the same oLFSR output sequence of length $2^n - 1$ is randomly shifted by z-bits with probability 50%, resulting in a change of bit 1 or 0 to 0 or 1, respectively, the quality of optical keystream and, thus, high level of security can be only reached, if $L_{\alpha_a} \to 1$, i.e., $C_{R_a} \to C$.
That contradicts the defined design requirements on oKG.

oKG with parallel LFSRs (oKG-B)

To avoid the repetition of the same bit sequence, such as in the case of oKG-A, while reducing the bit rate of pRNG, multiple oLFSRs can be used to generate an optical key. That would allow us a choice of different generator polynomials and different special sequences of maximal length $2^n - 1$ bits for the key generation. Fig. 5.3b shows such oKG implementation with P parallel oLFSRs and $1 : P$ optical switch. The resulting optical key K is generated with different oLFSRs that are randomly selected by the optical switch during the key generation process. Here, we propose to switch between parallel oLFSRs $\mathcal{I}$ times, i.e., perform oLFSR *reset*, and to reinitialize the switched oLFSRs with a new seed, i.e., perform oLFSR *reseed*. We refer to the time between two resets/reseeds as *reset cycle* t_c.

The pRNG-b initialized by control sequence k generates electronic bit sequence $M = [\tilde{n}_1, \tilde{p}_1, ..., \tilde{n}_j, \tilde{p}_j, ..., \tilde{n}_\mathcal{I}, \tilde{p}_\mathcal{I}]$ of length L_M at low bit rate C_{R_b} to define one out of P oLFSRs and to seed the same in any reset cycle. For any j^{th} reset cycle, $p = \lceil log_2(P) \rceil$ bits correspond to the index of the next oLFSR to be used, and sequence $\tilde{n}_j$ of length

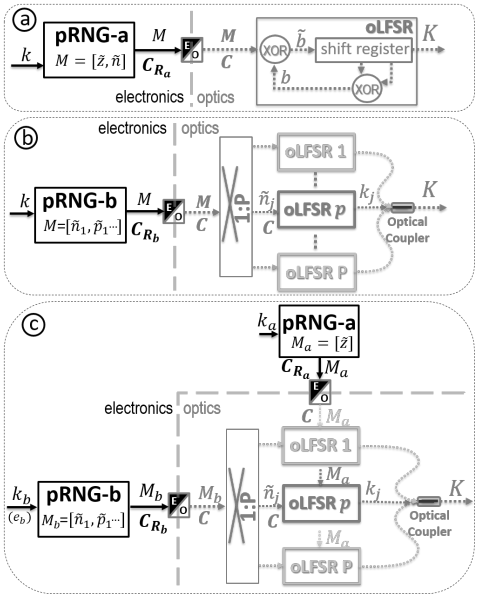


Figure 5.3: Engineering an optical key generator (oKG) [11]: a) Basic design, b) Parallel oLFSRs, c) Combination of a) and b).

n bits are used as oLFSR seed. Thus, the bit sequence $\tilde{p}_j$ configures optical switch to forward n bits into randomly selected oLFSR $\tilde{p}_j$. The seed-bits are converted by E/O converter bit by bit into optical impulses, just like in oKG-A. The generation of key K starts when the shift register of selected oLFSR p is filled by n initial bits, i.e., seeded. When the same oLFSR is randomly reselected by pRNG-b for the next reset cycle $j+1$, the generated seed $\tilde{n}_{j+1}$ can be used as a de-linearization stream to additionally shift the register of oLFSR without switching and interruptions of the key generation process.

Any oLFSR randomly selected by the switch generates only a part $k_j \equiv \alpha_b$ of an optical key K, where $L_{k_j} \equiv L_{\alpha_b} << 2^n - 1$, whereby $\mathcal{I}$ reset cycles are required to generate a whole key K, i.e.,

$K = [k_1, k_2, ..., k_j, ..., k_{\mathcal{I}}]$ and $L_K = L_m = \mathcal{I} \cdot L_{\alpha_b}$. Thus, the number of reset cycles required for generation of a whole optical key K is

$$\mathcal{I} = \frac{L_m}{L_{\alpha_b}}. \tag{5.3}$$

During any reset cycle t_c, pRNG-b with a bit rate C_{R_b} has to generate $n + p$ bits and the optical switch has to switch from the previous oLFSR to the newly selected one during time t_{sw}. During the same time t_c, an active oLFSR generates at rate C the key part α_b of length L_{α_b}. Thus, the duration of the reset cycle is

$$t_c = \frac{L_{\alpha_b}}{C} = \frac{p+n}{C_{R_b}} + t_{\mathrm{sw}}, \tag{5.4}$$

the required bit rate C_{R_b} of pRNG-b is then derived as

$$C_{R_b} = \frac{(p+n)C}{L_{\alpha_b} - t_{\mathrm{sw}}C}. \tag{5.5}$$

The implementation of oKG-B can be realized by considering $C_{R_b} << C$ and $L_{\alpha_b} << 2^n - 1$. Nevertheless, the probability that the same shifted bit pattern sequentially occurs within an optical key sequence K is now $(\frac{1}{P})^{\mathcal{I}}$, i.e., each time the same oLFSR is reselected. Let us consider the required bit rate of pRNG-b C_{R_b}, as defined by Eq. (5.5). The condition $C_{R_b} << C$ is only valid, when $(L_{\alpha_b} - t_{\mathrm{sw}}C) \to \infty$, i.e., $t_{\mathrm{sw}} \to 0$ and $L_{\alpha_b} \to \infty$, and, thus, $n \to \infty$, while the switching time t_{sw} needs to decrease with increasing optical line rate C. Although some optical switches are able to switch optical signals between output ports in nanoseconds, a faster switching time remains a challenge for the line rates beyond 100 Gbit/s. Thus, there can be a need for an oLFSR de-linearization technique, which can relax the requirement of the short switching time $t_{\mathrm{sw}} \to 0$. That is only possible, when the length L_{α_b} of any α_b can be increased without any increase in the oLFSR length n, i.e., $L_{\alpha_b} >> 2^n - 1$. However, that will result in a repetition of the same bit patterns within a key sequence, such as in the case of oKG-A, degrading a key quality as well as overall security.

Combination of oKG-A and oKG-B (oKG-C)

This method combines the previous two approaches to relax the requirement on the short switching time t_{sw} without repetition of the same bit pattern within an optical key K. Here, oKG-B is extended by pRNG-a with an adapted bit rate $C_{R_a} << C$. The resulting oKG-C (Fig. 5.3c) consists of two pRNGs, an optical switch and P parallel oLFSRs. Both pRNG-a and pRNG-b are initialized with secure sequences, k_a and k_b, respectively. The pRNG-b generates a sequence $M_b = [\tilde{n}_1, \tilde{p}_1, ..., \tilde{n}_{\mathcal{I}}, \tilde{p}_{\mathcal{I}}]$ at rate $C_{R_b} << C$, which contains p bits for random oLFSR selection and n bits of a new seed for any out of $\mathcal{I}$ reset cycles. The pRNG-a generates a sequence $M_a = [\tilde{z}]$ at rate $C_{R_a} << C$, i.e., z nonlinear bits, for nonlinear shifting of a randomly selected oLFSR $\tilde{p}_j$. The E/O converts the nonlinear bits of M_a and M_b into optical bits, which are forwarded towards the active oLFSR $\tilde{p}_j$. Thus, pRNG-a needs to generate at least one bit before selected oLFSR generated $L_{\alpha_a} \leq 2^n - 1$ bits requiring the bit rate C_{R_a} as defined by Eq. (5.1). Then, during the whole key generation process, pRNG-a generates z nonlinear bits as defined by Eq. (5.2). The combination of pRNG-a and pRNG-b assures that there is no repetition or shifting of the same bit patterns, which was the main weakness of oKG-A and oKG-B.

Due to oLFSR de-linearization with pRNG-a, the duration of reset cycle t_c and, thus, the number of bits L_{α_b} generated by a selected oLFSR before the next reset can be increased by a factor δ as follows

$$L_{\alpha_b} = \delta(2^n - 1), \tag{5.6}$$

where $\delta \in \mathbb{Q}$, $\delta > 0$. The bit rate of pRNG-b, which is used for oLFSR selection and seeding, and the number of required reset cycles are defined by Eqs. (5.5) and (5.3), respectively.

Let us now analyze the switching time t_{sw} of an optical switch. Based on the assumption $C_{R_b} << C$, Eq. (5.5) is only valid, if $0 < \frac{p+n}{L_{\alpha_b} - t_{\text{sw}} C} << 1$ and $L_{\alpha_b} - t_{\text{sw}} C > 0$. From the first constraint,

the switching time t_{sw} can be calculated as follows

$$t_{\text{sw}} << \frac{L_{\alpha_b} - p - n}{C}. \tag{5.7}$$

From the second constraint, we can derive $t_{\text{sw}} < \frac{L_{\alpha_b}}{C}$, which is not as strong as in Eq. (5.7), i.e., $\frac{L_{\alpha_b}-p-n}{C} < \frac{L_{\alpha_b}}{C}$. Thus, using Eq. (5.6), Eq. (5.7) can be modified as follows

$$t_{\text{sw}} < \frac{\delta(2^n - 1) - p - n}{C}. \tag{5.8}$$

As a result, the possible minimum value of δ as per system design can be derived from Eg. (5.8) as $\delta > \frac{t_{\text{sw}}C+p+n}{(2^n-1)}$.

Further De-linearization (oKG-D)

To additionally increase the entropy of the optical key, we need to increase the level of diffusion and confusion. To this end, oKG-C requires to provide variable system engineering parameters. Some components of proposed oKG-C, such as maximal bit rate of pRNGs and minimal switching time of optical $1 : P$ switch, are determined by current technology and cannot be altered. Other parameters such as a number of parallel oLFSRs P, its length n and minimal admissible bit rate of pRNGs are fixed by system design. On the other hand, the bit rate of pRNGs can be varied between the minimum admissible and maximum possible without impacting other fixed system parameters. Let us refer to these bit rate variations as *operational bit rate*. Then, the control plane can randomly define and periodically change the operational bit rates of pRNG-a and pRNG-b and, thus, the number of reset cycles $\mathcal{I}$ and the length of the sequence z, for a certain key generation session.

Let us assume that oKG-D has exactly the same architecture as oKG-C presented in Fig. 5.3c, but the pRNG-a and pRNG-b can provide $\mathcal{E}_a$ and $\mathcal{E}_b$ different bit rates, respectively. That can be realized, for instance, with an electronic traffic shaper and results in

$\mathcal{E}_a$ different possible lengths of M_a and in $\mathcal{E}_b$ possible lengths of M_b. Generally, there can be $\mathcal{E}_a = C_{\max_a} - C_{\min_a}$ and $\mathcal{E}_b = C_{\max_b} - C_{\min_b}$ different bit rates of pRNG-a and pRNG-b, respectively. Then, the maximal bit rate $C_{\max_a}$ and $C_{\max_b}$ are defined by current technology and the minimal admissible bit rates $C_{\min_a}$ and $C_{\min_b}$ are defined by the oKG design, i.e., by Eqs. (5.1) and (5.5), respectively. Thus, the minimum admissible bit rate of pRNG-a is derived as

$$C_{\min_a} \geq \frac{C}{L_{\alpha_a}} = \frac{C}{2^n - 1}, \tag{5.9}$$

with Eq. (5.5) and Eq. (5.6), the minimal admissible bit rate of pRNG-b can be determined as follows

$$C_{\min_b} \geq \frac{(p+n)C}{L_{\alpha_b} - t_{\mathrm{sw}}C} = \frac{(p+n)C}{\delta(2^n - 1) - t_{\mathrm{sw}}C}. \tag{5.10}$$

Generally, the bit rate of pRNGs can be selected once for a whole key generation session or periodically varied during, e.g., for any reset cycle. The level of diffusion and confusion increases with the increasing frequency of bit rate variation. The bit rate variation of pRNG-b for any reset cycle is only reasonable when the duration of the reset cycle can be changed. That would be, however, challenging because the optical switch would need to be reconfigured, which requires complex synchronization and control mechanisms. Consequently, the bit rate of pRNG-b should be randomly selected once, i.e., one out of all $\mathcal{E}_b$ possible, for the whole key generation session. In contrast, the variation of the bit rate of pRNG-a will only impact the length of the generated nonlinear sequence M_a and can be easily implemented in electronics without affecting optical components. Thus, we propose to randomly change the bit rate of pRNG-a for any reset cycle t_c. That process can be controlled, for instance, by an additional (third) pRNG initialized by the control plane.

We assume that the traffic shaper can vary the bit rate of pRNGs with tier of $1bit/s$, while the control plane sends control parameters

e_a and e_b, where $e_{a,b} = 1, 2, ..., \mathcal{E}_{a,b}$ to define the bit rates of the pRNGs. Then, the session bit rate of pRNG-(a,b) is determined as

$$C_{R_{a,b}}(e_{a,b}) = (e_{a,b} - 1) + C_{\min_{a,b}}. \tag{5.11}$$

The oKG-D receives three control parameters from the control plane, i.e., k_a, k_b and e_b, used for seeding of pRNG-a, pRNG-b and bit rate adaption of pRNG-b, respectively. The bit rate of pRNG-a is, on the other hand, periodically varied with a control parameter e_a.

5.5 An Information-Theoretical Approach for Security Analysis

This section presents an information-theoretical approach for analysis of a security level provided by previously designed oKGs.

5.5.1 Threat Model

Let us consider eavesdropping attacks on anonymous communication, whereby an attacker can access any optical link and encrypted data to reveal the secret message and to deanonymize source and destination. The assumption is that the attacker knows the length of secret data, the length of keys utilized, the structure (algorithm) and parameters of the cascaded oSC. Since standard pRNG in the cascaded oKG is the most cryptographically strong subsystem, it is assumed that the attacker can not break secure pRNG and can only try to guess its input and output sequences, i.e., parameters k, k_a, k_b, e_b and $\tilde{n}$, $\tilde{p}$, $\tilde{z}$, respectively. Thereby, the parameters k, k_a, k_b, e_b are from the control plane and assumed as strongly secure.

The optical key expansion based on oLFSRs is, in contrast, fundamentally insecure. Thus, an attacker can generally reveal the output sequence of pRNGs, i.e., parameters used for oLFSR initialization $\tilde{n}$, $\tilde{p}$ and $\tilde{z}$, by breaking the optical key expansion system. However, even if a set of initialization parameters can be broken at a time, the

attacker will not be able to derive the output sequence of a pRNG for any subsequent en-/decryption. Since the system is configured to use a certain optical key for en-/decryption only once, the attacker is forced to do the guessing work to break a key sequence. Thus, for any new en/decryption step, the attacker has to *guess* either the input or output bits of pRNGs. Moreover, we assume that the optical source data can only be revealed if the entire eavesdropped data is decrypted. The latter is a valid assumption because several steps of standard signal processing, e.g., source coding, interleaving, Forward Error Correction (FEC), scrambling, etc., are required before optical layered encryption. That makes it impossible for an attacker to detect the secret content or its parts without knowing the entire key and inverse steps of standard signal processing.

Classification of Guessing Attacks

Due to the assumptions of the threat model defined above, we use three definitions of guessing attacks based on information theory, i.e., Shannon's theory of secrecy and entropic security. We refer to the first type of attack as
(I) *Direct Guessing (DG) Attack*, whereby an attacker tries to decrypt the secret data with all possible key sequences without using the knowledge about cascade. In this case (Shannon's security), an attacker can only break encryption by guessing a secret message or a key, which is as long as the secret message. In the case of entropic security, the encryption key does not need to be as long as the secret message, and an attacker tries to guess a short secret key (further extended in any cipher). We refer to this type of attack as
(II) *Black Box (BB) Attack*, whereby an attacker utilizes the cascade as a black box and tries to guess its input parameters, i.e., generates bit sequences which are interpreted by oKG and pRNGs as valid input parameters. Additionally, the third category of guessing attacks is needed due to the insecure oLFSR-based subsystem of the

cascade, i.e., oSC. We define this new type of guessing attack as (III) *White Box (WB) Attack*[1], whereby the attacker uses knowledge about the cascaded system and tries to guess all input parameters of the insecure subsystem, i.e., oLFSR.

5.5.2 Security Analysis of optical Stream Cipher

Since the proposed oKG designed as a cascade adapts the length and a bit rate of the keystream, whereby the oLFSR generates multiple optical poor-entropy bits based on few high entropy bits from a pRNG, new security evaluation techniques are required. The security analysis should prove that (i) the insecure oLFSR-based subsystem does not degrade the security of the overall cascade, and (ii) the long optical keystream generated by oLFSRs provides enough uncertainty. Thus, our security analysis presents an information-theoretical evaluation of the computational complexity of a successful guessing attack, i.e., analysis of uncertainty (entropy and equivocation) of optical encryption key generated by cascaded oKG.

Definition 5.1 *A cascade-based encryption system that is unbreakable with any guessing attack within any reasonable time is referred to as* **computationally secure**.

Following principles of *entropic security*, a secret message m of length L_m bits is an arbitrary bit sequence out of all 2^{L_m} possible sequences with the highest entropy defined as $H(m) = L_m$. When m was encrypted with a key K into message $K(m)$ of the same length L_m, an eavesdropper who accessed the encrypted message $K(m)$ tries to decrypt it applying guessing work, i.e., attempts to gain message m by guessing of encryption key K. To evaluate computational security, the computational complexity of a guessing attack is analyzed

[1] Note that the white-box adversary was introduced in 2003 by Chow et al. [133, 134] in the context of attacks on cryptographic software. Here, this term is redefined and used in the context of guessing attacks on oKG.

in a number of bits, which have to be guessed by an attacker. That number of bits can be defined by *equivocation* $H(m|K(m))$.

Disclosed Optical Bits

Without loss of generality, an encrypted optical data $K(m)$ can leak some information of ℓ bits about message m reducing equivocation $H(m|K(m))$ and, thus, the computational complexity of guessing. However, due to standard signal processing steps before E/O conversion, the optical data m is not readable for an attacker already before optical en-/decryption and, thus, the optical key K can generally be a sequence of only ones or only zeros. Consequently, there can be 2^{L_m} different possible keys. On the other hand, XOR encryption with the optical key, which contains ℓ consecutive ones or zeros, will result in ℓ inversed bits or ℓ unchanged bits within m, where $\ell \geq 0$. These ℓ inverted or unchanged bits can disclose the secret content of m or its parts particularly in the last optical anonymization node and, thus, reduce the computational complexity of guessing attacks.

Following principles of entropic security [117, 116], let us first analyze the amount of disclosed optical bits ℓ of secret message. Without loss of generality, we consider oKG as a binary Markov source presented by two states and four transitions between these states. Then, the probability of i consecutive ones or zeros generated by oKG can be evaluated as follows

$$p(i) = (1 - p_{00/11})p_{00/11}^{i-1}, \tag{5.12}$$

where $p_{00/11} < 1$ is the one-step transition probability to return in the same state, i.e., to generate "1" or "0" after generated "1" or "0", respectively. Thus, the expected length of disclosed consecutive bits is defined as $\bar{\ell} \approx \frac{1}{1-p_{00/11}}$ bits, if $L_m \to \infty$. On the other hand, an attacker can interpret the secret message based on s *symbols* represented by multiple consecutive bits, e.g., $s = 1$ symbol is represented by $L_s = 16$ bits in usual hexadecimal computer systems.

Thus, the attacker needs whole symbols to interpret a part of the secret message, whereby ℓ_s/L_s consecutive symbols can be leaked to the attacker. Then, the expected length of disclosed consecutive bits, which represent a sequence of symbols, is determined as follows

$$\begin{aligned}\ell_s = \sum_{i=0}^{L_m} \left\lfloor \frac{i}{L_s} \right\rfloor L_s p(i) &= \sum_{i=0}^{L_m} \left\lfloor \frac{i}{L_s} \right\rfloor L_s (1 - p_{00/11}) p_{00/11}^{i-1} \\ &< \frac{1}{1 - p_{00/11}},\end{aligned} \tag{5.13}$$

where $\left\lfloor \frac{i}{L_s} \right\rfloor$ determines the maximum number of disclosed symbols. As a result, the oKG needs to be adjusted to provide $p_{00/11} \leq 0.5$ to leak $\ell_s < L_s$. Generally, if an attacker interprets s symbols of message m, these symbols do not need to be consecutive. For instance, if $s = 3$, three symbols can be distributed within secret data sequence m in the following fashion: 3 consecutive symbols or 2 consecutive symbols and one separate symbol or 3 separate symbols and, thus, result in different occurrence probabilities. Then, the probability of 3 disclosed symbols, i.e., $\ell_s = 3L_s$, is $\tilde{p}(3) = p(3L_s) + p(2L_s)p(L_s) + p(L_s)^3$. Using Eq. (5.12), let us define a probability $p'(y) = p(yL_s) = (1 - p_{00/11}) p_{00/11}^{yL_s - 1}$ that y symbols out of s disclosed symbols can occur consecutively. Thus, we can represent the probability of 3 disclosed symbols from the example above as $\tilde{p}(3) = p'(3) + p'(2)p'(1) + p'(1)^3$. Based on this simple example, the occurrence probability $\tilde{p}(s)$ of s disclosed symbols is given by

$$\begin{aligned}\tilde{p}(s) = p'(1)^s \mathfrak{k}_0 + \sum_{\substack{i=2, \\ \text{if } s \geq 2}}^{s} \sum_{j_0=0}^{\mathcal{G}_0} p'(i)^{\chi_0 - j_0} \sum_{\substack{j_1=0, \\ \text{if } \chi_1' \geq i-1}}^{\mathcal{G}_1} p'(i-1)^{\chi_1 - j_1} \ldots \\ \ldots \sum_{\substack{j_{s-1}=0, \\ \text{if } \chi_{s-1}' \geq i-s+1}}^{\mathcal{G}_{s-1}} p'(i-s+1)^{\chi_{s-1} - j_{s-1}} \cdot \mathfrak{k}_1,\end{aligned} \tag{5.14}$$

where the variables $\mathfrak{k}_0$ and $\mathfrak{k}_1$ are logical variables to prove if a certain

combination of consecutive and separated symbols can be arranged within secret message m of length L_m. For the above-mentioned example, the allocation of three separate symbols out of $s = 3$ disclosed symbols within data of 4 symbols would not be possible. That is because each out of 3 disclosed symbols needs to be separated by at least one undisclosed symbol, i.e., at least 2 additional symbols. Thus, the message length needs to have at least 5 symbols. Thus, $\mathfrak{k}_0{=}1$ if and only if $s+(s-1)\leq\frac{L_m}{L_s}$; and $\mathfrak{k}_1{=}1$ if and only if a certain symbol combination can be arranged within data resulting in $\sum_{\epsilon=0}^{s-1}(i-\epsilon)(\chi_\epsilon-j_\epsilon)+(\chi_\epsilon-1)+\mathfrak{k}_3(\epsilon)\leq\frac{L_m}{L_s}$, where the third logical variable $\mathfrak{k}_3(\epsilon){=}1$ if and only if all s symbols are not separated, i.e., $\mathfrak{k}_0 = 0$, and the certain combination of $(i-\epsilon)$ consecutive symbols exists, i.e., $(i-\epsilon)(\chi_\epsilon-j_\epsilon)>0$. Moreover, the certain allocation of s symbols is possible, but not all s symbols are located considering a certain ϵ, i.e., $\sum_{\nu=0}^{\epsilon}(i-\nu)(\chi_\nu-j_\nu)<s$. The first addend in Eq. (5.14) determines the probability that all s symbols are separated from each other. The variable i defines any other possible position of s disclosed symbols within secret optical data, whereby at most i, $2\leq i\leq s$, consecutive symbols can be differently allocated within the optical data. Any ϵ, where $0\leq\epsilon\leq s-1$, determined through j_ϵ affects the probability $\tilde{p}(s)$ if there are at least $i-\epsilon$ remaining symbols out of all s disclosed symbols, which were not considered by the previous sums. The amount of unconsidered symbols is determined as function χ'_ϵ and is calculated for any sum ϵ as follows

$$\chi'_\epsilon \equiv \chi'_\epsilon(j_0, j_1, ..., j_{\epsilon-1}) = s - \sum_{\delta=0}^{\epsilon-1}(\chi_\delta - j_\delta)(i-\delta), \tag{5.15}$$

where $\chi_\delta \equiv \chi_\epsilon$ is the maximum number of symbol combinations, which consist of $(i-\epsilon)$ consecutive symbols, positioned separately from each other and is determined as follows

$$\chi_\epsilon = \chi_\epsilon(j_0, j_1, ..., j_{\epsilon-1}) = \begin{cases} \left\lfloor \frac{s}{i} \right\rfloor, & \text{if } \epsilon = 0, \\ \left\lfloor \frac{\chi'_\epsilon}{i-\epsilon} \right\rfloor, & \text{else,} \end{cases} \tag{5.16}$$

where s is the total number of disclosed symbols and i is the maximum number of consecutive symbols in a certain symbol combination as defined by a certain state, i.e., $j_0, j_1, ..., j_{\epsilon-1}$, in Eq. (5.14). However, the minimum number of combinations of $i-\epsilon$ consecutive symbols located separately within secret data sequence can be calculated as $\chi_\epsilon - \mathcal{G}_\epsilon \geq 0$, whereby $\mathcal{G}_\epsilon$ is a function of unconsidered symbols χ'_ϵ and separates symbol combinations of $(i-\epsilon)$ consecutive symbols, i.e., χ_ϵ, and is determined as follows

$$\mathcal{G}_\epsilon = \begin{cases} \chi_0 - 1, & \text{if } \epsilon = 0, \\ min\left\{\chi_\epsilon; max\left\{\left\lfloor \frac{\chi'_\epsilon}{i-\epsilon-1} \right\rfloor - 1; 0\right\}\right\}, & \text{if } \epsilon > 0 \text{ and} \\ & (i-\epsilon-1) \geq 1, \\ 0, & \text{else.} \end{cases} \tag{5.17}$$

Thus, the mean number of disclosed bits, which can be interpreted by an attacker symbol-by-symbol is given as follows

$$\ell = \sum_{i=0}^{L_m} \left\lfloor \frac{i}{L_s} \right\rfloor \tilde{p}\left(\left\lfloor \frac{i}{L_s} \right\rfloor \right). \tag{5.18}$$

Security Analysis in the Case of Guessing Attacks

Next, we evaluate three different guessing attacks on oSC: DG, BB and WB, which are characterized by different equivocation, i.e., $H_{DG}(m|K(m))$, $H_{BB}(m|K(m))$, $H_{WB}(m|K(m))$, respectively. Since the attacker knows the required input of secure subsystem (pRNG), encryption algorithm and internal parameters of oKG, the attacker can select those attack, which has the minimal computational complexity, i.e., which requires a minimal number of bits to be guessed, $min\{H_{DG}(m|K(m)), H_{BB}(m|K(m)), H_{WB}(m|K(m))\}$.

Let us assume that an attacker collected encrypted data m_i on the input interface of an anonymization node A_i and tries to remove

one encryption layer, i.e., to gain data m_{i+1}. Generally, the probability that the attacker can compromise m_{i+1}, i.e., $Pr(m_{i+1}|m_i)$, is a probability to find a related optical key K_i, i.e., $Pr(m_{i+1}|m_i) = Pr(K_i|m_i)$, i.e., $H(m_{i+1}|m_i) \equiv H(K_i|m_i)$. As previously defined, the input parameters, i.e, k, k_a, k_b, e_b, of oKG are securely generated and distributed, i.e., have a high level of entropy, and can only be revealed by random guessing. We refer to these parameters as *strongly secure parameters*. Thus, an attacker has to (I) guess an optical key K, i.e., DG; (II) perform BB attack by guessing strongly secure parameters; or (III) WB attack by guessing the input parameters of the oLFSR based subsystem, i.e., the output of pRNGs.

Direct Guessing (DG) Attack

The key equivocation observed by an attacker, who tries to guess the optical key K_i used by anonymization node A_i, is independent of oKG design or its control parameters and is determined as

$$H_{DG}(K_i|m_i) = log_2(2^{L_m-\ell}) = L_m - \ell, \tag{5.19}$$

whereby the attacker can directly guess message m_{i+1} with similar computational complexity, i.e., $H_{DG}(m_{i+1}|m_i) = L_m - \ell$.

Black Box (BB) Attack

The oKG-A consists only of one pRNG-a that is seeded by a strongly secure control parameter k of length L_k. Thus, only k needs to be guessed by the attacker resulting in equivocation determined as

$$H^{\mathrm{A}}_{BB}(K_i|m_i) = log_2\left(2^{L_k}\right) = L_k. \tag{5.20}$$

The oKG-B contains one pRNG-b that is seeded by strongly secure control parameter k of length L_k bits arrived from the control plane. Thus, the attacker has to guess only this parameter k out of 2^{L_k} possible, resulting in the same equivocation as defined by Eq. (5.20).

The oKG-C is a combination of oKG-A and oKG-B and contains pRNG-a and pRNG-b, which are separately seeded by strongly secure control parameters k_a and k_b from the control plane, respec-

tively. In case of BB attack, the attacker has to guess both parameters, i.e., k_a and k_b of length L_{k_a} and L_{k_b} bits and, thus, $L_{k_a} + L_{k_b}$ bits totally. Thus, the attacker has to guess k_a and k_b out of all $2^{L_{k_a}+L_{k_b}}$ bit sequences, resulting in the equivocation defined as

$$H_{BB}^{\mathrm{C}}(K_i|m_i) = log_2\left(2^{L_{k_a}+L_{k_b}}\right) = L_{k_a} + L_{k_b}. \tag{5.21}$$

The oKG-D is an extension of oKG-C and consists of the same system components. The main difference, however, is that pRNGs-a and pRNG-b have variable bit rates. Here, an attacker has to guess strongly secure parameters k_a, k_b and e_b, i.e, $L_{k_a} + L_{k_b} + L_{e_b}$ bits out of $2^{L_{k_a}+L_{k_b}+L_{e_b}}$, where $L_{e_b} = log_2(\mathcal{E}_b)$ resulting in

$$H_{BB}^{\mathrm{D}}(K_i|m_i) = log_2\left(2^{L_{k_a}+L_{k_b}+L_{e_b}}\right) = L_{k_a} + L_{k_b} + L_{e_b}. \tag{5.22}$$

White Box (WB) Attack

The oKG-A consists of pRNG-a and one oLFSR of length n. Thus, the attacker tries to guess the internal parameters of oKG-A, i.e., output bit sequence M from pRNG-a, which consists of n seed bits and z bits for oLFSR shifting. Thus, the equivocation is defined as

$$H_{WB}^{\mathrm{A}}(K_i|m_i) = log_2\left(2^{L_M}\right) = log_2\left(2^{n+z_\ell}\right) = n + z_\ell, \tag{5.23}$$

where $z_\ell = \frac{L_m - \ell}{L_{\alpha_a}}$ according to Eq. (5.2).

The oKG-B consists of an optical switch, pRNG-b and P parallel oLFSRs of length n. The switch uses sequence M from pRNG-b to randomly define one oLFSR for key generation in the next reset cycle. The pRNG-b generates nonlinear bits $M = [\tilde{n}_1, \tilde{p}_1, ..., \tilde{n}_\mathcal{I}, \tilde{p}_\mathcal{I}]$, where $\tilde{p}_j$ of length $p = log_2(P)$ is the index of oLFSR for the next reset cycle j and $\tilde{n}_j$ of length n is a seed. The attacker uses the knowledge about oKG-B, i.e., the number of resets $\mathcal{I}$, bit rate of pRNG-b C_{R_b}, the switching time t_{sw}, the number of oLFSRs P and its length n as well as the line rate C, which are constant for any encryption. To generate a correct decryption key K_i, the attacker

needs to guess the internal parameters of oKG-B, i.e., nonlinear bit sequences M_ℓ of length L_{M_ℓ} generated by pRNG-b, respectively. Here, $L_{M_\ell} = \mathcal{I}_\ell(n+p)$ and, from Eq.(5.3), $\mathcal{I}_\ell = \frac{L_m - \ell}{L_{\alpha_b}}$. Another option, however, is to guess oLFSR-seed combination for each out of I_ℓ reset cycles. Then, the key equivocation can be calculated as

$$H^{\mathrm{B}}_{WB}(K_i|m_i) = log_2(2^n P)^{\mathcal{I}_\ell} = \mathcal{I}_\ell(n+p) = log_2 2^{L_{M_\ell}} = L_{M_\ell}. \tag{5.24}$$

The oKG-C consists of pRNG-a and pRNG-b, P oLFSRs of length n and optical switch. The pRNG-b generates sequence $M_b = [\tilde{n}_1, \tilde{p}_1, ..., \tilde{n}_\mathcal{I}, \tilde{p}_\mathcal{I}]$, i.e., defines oLFSR for the next reset cycle with p bits and its n bit seed. The pRNG-a generates a sequence M_a of length z for oLFSR de-linearization. Using the knowledge about the oLFSR-based subsystem, i.e., the number of resets $\mathcal{I}_\ell$, bit rates C_{R_a} and C_{R_b}, switching time t_{sw}, number of oLFSRs P and its length n as well as line rate C, an attacker needs to guess the internal parameters of oKG-C generated by pRNGs. Thus, the attacker tries to guess nonlinear bit sequences M_{a_ℓ} and M_{b_ℓ} of length $L_{M_{a_\ell}} = z_\ell$ and $L_{M_{b_\ell}} = \mathcal{I}_\ell(n+p)$, respectively, resulting in key equivocation:

$$H^{\mathrm{C}}_{WB}(K_i|m_i) = log_2\left(2^{L_{M_{a_\ell}} + L_{M_{b_\ell}}}\right) = \mathcal{I}_\ell(n+p) + z_\ell. \tag{5.25}$$

In case of WB attack on oKG-D, an attacker has to guess values of internal parameters generated by pRNG-a and pRNG-b as well as the session bit rates of pRNGs, i.e., $C_{R_a}(e_a)$ and $C_{R_b}(e_b)$ out of $\mathcal{E}_a$ and $\mathcal{E}_b$ possible, respectively. Now, the amount of reset cycles is variable and depends on the session bit rate $C_{R_b}(e_b)$ of pRNG-b. Thus, an attacker has to consider all $\mathcal{E}_b$ possible numbers of resets for all $\mathcal{E}_b$ possible bit rates and all possible seed-oLFSR combinations for any reset cycle. Then, the attacker has to guess a bit rate of pRNG-a $C_{R_a}(e_a)$ out of $\mathcal{E}_a$ possible for any reset cycle and corresponding sequence M_{a_ℓ} out of all $2^{z_\ell(\mathcal{E}_a, e_b)}$ possible. Then, the key

equivocation of the key generated by oKG-D is determined as

$$\begin{aligned} H^{\mathrm{D}}_{WB}(K_i|m_i) &= \sum_{e_b=1}^{\mathcal{E}_b} log_2\left(\left(2^{z_\ell(\mathcal{E}_a,e_b)}2^n P\right)^{\mathcal{I}_\ell(e_b)}\right) = \\ &= \sum_{e_b=1}^{\mathcal{E}_b} \mathcal{I}_\ell(e_b)(n+p+z_\ell(\mathcal{E}_a,e_b)), \end{aligned} \tag{5.26}$$

where the number of resets $\mathcal{I}_\ell(e_b)$ can be derived from Eqs. (5.3) and (5.4) considering ℓ as follows

$$\mathcal{I}_\ell(e_b) = \frac{(L_m-\ell)C_{R_b}(e_b)}{\left(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\right)C}. \tag{5.27}$$

Since the pRNG-a generates bits with the same bit rate as long as selected oLFSR $\tilde{p}_j$ generates L_{α_b} bits of optical key K_i, an attacker needs to consider all possible bit rates $\mathcal{E}_a$ of pRNG-a and to test $z_\ell(\mathcal{E}_a, e_b)$ bits generated by pRNG-a for any reset cycle. In general, to break a key K_i, it is not enough to generate all possible sequences $M_{a_\ell} = [\tilde{z}_\ell(\mathcal{E}_a, e_b)]$ and to test them. An attacker has to test all possible bit rates of pRNG-a for each reset cycle because the bit rate with that bits of the sequence M_{a_ℓ} are injected into the oLSFR at least as important as the value of the sequence M_{a_ℓ}. With Eqs. (5.1), (5.2) and (5.11), $z_\ell(\mathcal{E}_a, e_b)$ can be determined as follows

$$\begin{aligned} z_\ell(\mathcal{E}_a,e_b) &= \sum_{e_a=1}^{\mathcal{E}_a} \frac{L_{\alpha_b}(e_b)C_{R_a}(e_a)}{C} \\ &= \frac{\mathcal{E}_a}{C}\left(\frac{\mathcal{E}_a-1}{2}+C_{\min_a}\right)L_{\alpha_b}(e_b), \end{aligned} \tag{5.28}$$

where $L_{\alpha_b}(e_b)$ depends on the session bit rate of pRNG-b $C_{R_b}(e_b)$ and can be derived from Eq. (5.4) as follows

$$L_{\alpha_b}(e_b) = \frac{C\left(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\right)}{C_{R_b}(e_b)}. \tag{5.29}$$

5.5.3 A Comparative Security Analysis of AES

AES provides an iterative approach, which is a sequence of simpler cryptographic primitives (rounds) to the input data and symmetric key k'. None of these primitives alone would be able to provide cryptographic security, but a combination and repetition of them achieve the level of diffusion and confusion required in today's cryptosystems, i.e., strong computational security [31, 95, 135]. For easier understanding, we denote in this section the parameters from Sec. 5.8 with an apostrophe but use the same meaning of the symbols.

Before AES encryption begins, any 16 bytes of source data are logically arranged in a 4x4 matrix and bytes of encryption key build a matrix with 4 rows and 4, 6 or 8 columns, when key length $L_{k'}$ is set to 128, 192 or 256 bits, respectively. Since the source data is usually larger than 16 bytes, the encryption of it requires multiple encryption cycles. At the beginning of any encryption cycle, the 16 bytes of the original key k' or key from the previous cycle is byte by byte XOR concatenated with the data matrix. The result of this concatenation and the key itself are then iteratively transformed during multiple encryption rounds. To this end, we can consider a chain of encryption rounds and encryption cycles as a cascade. Each AES round is composed of a set of four basic byte operation steps on data matrix: 1) byte substitution using substitution table; 2) shifting rows based on a fixed permutation; 3) mixing columns based on modulo multiplications with constants and 4) adding a round key to data with XOR operation.

To complete step 4) of any round i (adding a round key), the round key k'_i needs to be generated from the original key k' or the round key k'_{i-1} of the previous round requiring *key expansion*. Any key K'_j in any encryption cycle j can be presented as a sequence of round keys, k'_0, k'_1, ..., k'_i, ..., $k'_{R_{k'}}$, and noted as $K' = [k'_0, k'_1, ..., k'_i, ..., k'_{R_{k'}}]$, where $R_{k'}$ is a number of encryption rounds required to encrypt data matrix of 16 bytes, i.e., $L_{K'_j} = 128R_{k'}$ bits. The resulting

encryption key K' is presented as $[K'_1, ..., K'_j, ..., K'_c]$ and has length $L_{K'} = cL_{K'_j}$, where $c = \frac{L_{m'}}{128}$ is a number of encryption cycles.

The AES decryption consists of inverse steps. Although the order in which the four operation steps are executed is different for en- and decryption, the number of byte operations required is the same resulting in the same computational complexity.

A secret message m' of length $L_{m'}$ bits is an arbitrary bit sequence with the entropy defined as $H'(m') = L_{m'}$. An attacker, who accessed encrypted message m'_i on input interface of anonymization node A_i, tries to remove one encryption layer by performing guessing work, i.e., tries to gain a message m'_{i+1}. We consider the probability that the attacker can compromise m'_{i+1}, i.e., $Pr(m'_{i+1}|m'_i)$, as a probability to find a related key K'_i for AES algorithm, i.e., $Pr(m'_{i+1}|m'_i) = Pr(K'_i|m'_i)$ and, thus, $H'(m'_{i+1}|m'_i) \equiv H(K'_i|m'_i)$.

The AES key k' can generally consist of only ones or only zeros resulting in $2^{L_{k'}}$ possible key sequences. These two specific key sequences do not significantly degrade AES security level because they are transformed into a sequence of ones *and* zeros already after the first key expansion round, i.e., $\ell = 0$.

An attacker can not access the strongly secure AES key k' and, thus, has to (I) guess the message m'_{i+1} out of all $2^{L_{m'}}$ possible, i.e., DG attack; (II) perform BB attack by guessing key k'; or (III) WB attack by guessing the internal parameters of the AES system, i.e., any round key in any encryption cycle.

Direct Guessing (DG) Attack

When AES algorithm is used, the attacker can try to directly guess the message m'_{i+1}. With AES algorithm, there is no generally valid key of length $L_{m'}$, which can decrypt m'_i by one transformation, e.g., with XOR operation, i.e., the equivocation in case of DG attack is

$$H'_{DG}(m'_{i+1}|m'_i) = log_2(2^{L_{m'}}) = L_{m'}. \tag{5.30}$$

Considering Eqs. (5.30) and (5.19), if $m \equiv m'$ and $L_m - \ell = L_{m'}$ then $H_{DG}(K_{i+1}|m_i) = H'_{DG}(m'_{i+1}|m'_i)$. Thus, an attacker, who performs

DG attack to compromise a secret message, has to guess the same number of bits independent of the encryption method, oSC or AES.

Black Box (BB) Attack

In the case of BB attack, an attacker has to guess a strongly secure AES key k' with entropy $H'(k') = H'_{BB}(k'_i|m'_i)$ and, then, utilize them for AES decryption. When AES were compromised, an attacker would try every encryption key k' out of $2^{L_{k'}}$ possible to decrypt an eavesdropped message with a well-known algorithm. Thus, the equivocation of the AES key can be determined as follows

$$H'_{BB}(k'_i|m'_i) = log_2\left(2^{L_{k'}}\right) = L_{k'}. \tag{5.31}$$

Corollary 5.1 *To reach the same universally accepted level of security, such as provided by AES, the control sequence utilized by any oSC, i.e., pRNGs, needs to be at least as long as the AES key k'.*

White Box (WB) Attack

In case of WB attack, an attacker will use the knowledge about the subsystems of a cascade, i.e., AES algorithm for any encryption round, to compromise the secret message m'_{i+1}. The attacker knows the length of the secret message and the key, i.e., $L_{m'}$ and $L_{k'}$, respectively. Thus, he knows the number of decryption cycles c required to decrypt the whole message m'_i and the number of decryption rounds per decryption cycle $R_{k'}$. The number of AES encryption rounds $R_{k'}$ performed on any 16 data bytes (4x4 matrix) depends on the key length $L_{k'}$ and is defined as $R_{k'} = 6 + \frac{L_{k'}}{32}$. For a fair comparison, we assume, such as for the oSC, that the attacker has to decrypt a whole message m'_i, i.e., all $L_{m'}$ bits, to be able to decide if the guessed key is the correct one. As a result, the attacker attempts to guess any 128 bits of any possible original key and any round key used in each decryption round and each decryption cycle.

That results in the following equivocation

$$H'_{WB}(K'_i|m'_i) = log_2\left(\left(2^{128(R_{k'}+1)}\right)^{\frac{L_{m'}}{128}}\right) = (R_{k'}+1)L_{m'}. \tag{5.32}$$

5.5.4 Creating Security Benchmark

Ideally, any encryption system follows the general condition for a *perfect secrecy* defined as $H(m|K(m)) = H(m) = L_m$ and $\ell = 0$, while that system needs to satisfy the condition $min\{H_{DG}(m|K(m)), H_{BB}(m|K(m)), H_{WB}(m|K(m))\} \geq L_m$. However, standard ciphers (e.g., AES) as well as the proposed oSC, e.g., based on oKG-D, cannot provide perfect secrecy. That is due to the fact that the length of the source message is generally much larger than the input parameters of the cryptosystem, e.g., $L_{k_a} + L_{k_b} + L_{e_b} << L_m$ for oKG-D. Based on this fact, we analyze next the capability of oSC to be *computationally secure* and define two types of cascaded systems:

Definition 5.2 *A cascade with a minimal computational complexity of guessing defined by the computational complexity of BB attacks is considered as* ***computationally secure cascade (CSC)****, i.e.,* $H_{WB}(m|K(m)) \geq H_{BB}(m|K(m)) \cap H_{DG}(m|K(m)) \geq H_{BB}(m|K(m))$.

Definition 5.3 *A computationally secure cascade with a maximal computational complexity of guessing defined by the computational complexity of WB attack, is referred to as* ***perfectly balanced cascade (PBC)****, i.e.,* $H_{WB}(m|K(m)) \geq H_{DG}(m|K(m)) \geq H_{BB}(m|K(m))$.

Thus, an attacker uses the BB attack for CSC due to the smallest computational complexity of guessing, whereby he guesses the strongly secure input parameters of the secure subsystem, i.e., pRNGs in oKG. In the case of PBC, an attacker would guess the entire key

or the strongly secure input parameters of cascade with less computational complexity than in the case of guessing of internal system parameters, e.g., of insecure oLFSR based subsystem in oKG.

In regard to oKG-A, using following Def. 5.2 and Eqs. (5.19), (5.20) and (5.23), the conditions for computationally secure oKG-A, i.e., $L_m - \ell \geq L_k$ and $n + z_\ell \geq L_k$ or, $n + \frac{L_m - \ell}{L_{\alpha_a}} \geq L_k$, are valid, if $L_m - \ell \to \infty$ and $L_{\alpha_a} << L_m - \ell$, $L_k << L_m - \ell$. That is always the case for optical networks and oKG-A, if $\ell << L_m$. In contrast, oKG-A can not be implemented as a PBC as per Def. 5.3, as a condition for that would be $n + z_\ell \geq L_m - \ell \geq L_k$ and is only valid if pRNG-a generates at least as many nonlinear bits $(n + z_\ell)$ as many key bits were generated by oLFSR $(L_m - \ell)$. That will, however, requires pRNG-a operates at optical line rate and result in no need for oLFSR, i.e., contradicts our assumptions $C_{R_a} << C$ and $L_M << L_m$. A pRNG implemented on the electronic layer (hardware or software) can currently operate at bit rates up to 1.3 Gb/s [136]. In contrast, the optical line rate is much larger, up to 400 Gb/s. Thus, to generate a long key sequence at a line rate, there is a need for a large number of pRNGs, which is an impractical solution.

With Def. 5.2, the condition for computationally secure oKG-B can be derived with Eqs. (5.19), (5.20) and (5.24) as $L_m - \ell \geq L_k$ and $\mathcal{I}_\ell(n + p) \geq L_k$, if $L_m - \ell \to \infty$ and $L_{\alpha_a} << L_m - \ell$, $L_k << L_m - \ell$, which are valid. Thus, oKG-B is CSC by design. However, the condition for perfectly balanced oKG-B from Def. 5.3 is $\mathcal{I}_\ell(n + p) \geq L_m - \ell \geq L_k$. That condition can be modified as $\frac{(L_m - \ell)(n+p)}{L_{\alpha_b}} \geq L_m - \ell$ and, thus, $(n + p) \geq L_{\alpha_b}$. Since $(n + p)$ bits are generated by pRNG-b and L_{α_b} bits are generated by oLFSR any reset cycle, oKG-B is only PBC, if pRNG-b generates at least as many nonlinear bits $(n + p)$ as many key bits were generated by oLFSR $\tilde{p}_j$, i.e., L_{α_b} bits. That will, however, require that pRNG-b operates at optical line rate and result in no need for oLFSRs, i.e., contradicts our assumptions $C_{R_b} << C$ and $n + p << L_{\alpha_b}$

($L_M << L_m$). Thus, oKG-B can not be designed as PBC.

The condition for computationally secure oKG-C from Def. 5.2 is valid, i.e., $\mathcal{I}_\ell(n+p) + z_\ell \geq L_{k_a} + L_{k_b}$. With Eqs. (5.3) and (5.2), $\frac{(L_m-\ell)(n+p)}{L_{\alpha_b}} + \frac{L_m-\ell}{L_{\alpha_a}} \geq L_{k_a} + L_{k_b}$, if $L_m - \ell \to \infty$, $L_{\alpha_a} << L_m - \ell$, $L_{\alpha_b} << L_m - \ell$ and $L_{k_a} + L_{k_b} << L_m - \ell$, that is always valid in optical networks with oSC based on oKG-C, if $\ell << L_m$. Based on Def. 5.3, the proposed oKG-C is PBC, if $H^{\mathrm{C}}_{WB}(K_i|m_i) \geq H^{\mathrm{C}}_{DG}(K_i|m_i) \geq H^{\mathrm{C}}_{BB}(K_i|m_i)$, $\mathcal{I}_\ell(n+p) + z_\ell \geq L_m - \ell \geq L_{k_a} + L_{k_b}$, if pRNG-a and pRNG-b generate at least as many bits, i.e., M_a and M_b, as many bits oLFSR generates, i.e., $\mathcal{I}(n+p) + z \geq L_m$, $\ell = 0$. That contradicts the requirements on oLFSRs to 1) expand the bits from pRNGs into longer sequences and 2) adapt key's bit rate to optical line rate, i.e., oKG-C can not be designed as PBC.

Lemma 5.1 *The oKG-D with variable bit rates of pRNGs is a perfectly balanced cascade if the number of different bit rates of pRNG-a is determined as follows*

$$\mathcal{E}_a \geq \sqrt{\frac{(2C_{min_a}-1)^2}{4} - \frac{2(n+p)\sum\limits_{e_b=1}^{\mathcal{E}_b} \frac{C_{R_b}(e_b)}{\left(p+n+t_{sw}C_{R_b}(e_b)\right)} - 2C}{\mathcal{E}_b}} - \\ - \frac{(2C_{min_a}-1)}{2}. \tag{5.33}$$

The proof of Lemma 5.1 is presented in Appendix 7.9.

With Eqs. (5.30), (5.31) and (5.32), the condition for AES as computationally complex and perfectly balanced cascade $H'_{WB}(K'_i|m'_i) \geq H'_{DG}(m'_{i+1}|m'_i) \geq H'_{BB}(k'_i|m'_i)$ is valid, i.e., $(R_{k'}+1)L_{m'} \geq L_{m'} \geq L_{k'}$. Thus, the AES encryption system is PBC for any standardized size of key k' and any length of secret message m'.

Table 5.1 summarizes the resulting key equivocation related to guessing attacks considered, each proposed oKG design and, for comparison, of AES. Here, we would like to note that oKG-A, oKG-B, and oKG-C can provide high computational security despite their

	DG attack $H_{DG}(K_i\|m_i)$	BB attack $H_{BB}(K_i\|m_i)$	WB attack $H_{WB}(K_i\|m_i)$	Security Evaluation
AES	$L_{m'}$	$L_{K'}$	$(R_{K'}+1)L_{m'}$	PBC
oKG-A	$L_m - \ell$	L_k	$n + z_\ell$	CSC
oKG-B	$L_m - \ell$	L_k	$\mathcal{I}_\ell(n+p)$	CSC
oKG-C	$L_m - \ell$	$L_{k_a} + L_{k_b}$	$\mathcal{I}_\ell(n+p) + z_\ell$	CSC
oKG-D	$L_m - \ell$	$L_{k_a} + L_{k_b} + L_{e_b}$	$\sum_{e_b=1}^{\mathcal{E}_b} \mathcal{I}_\ell(e_b)(n+p+z_\ell(\mathcal{E}_a, e_b))$	PBC

Table 5.1: Security Evaluation of AES and oKGs [11].

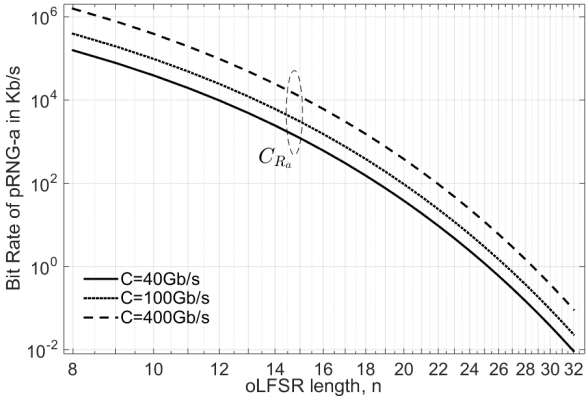


Figure 5.4: A minimal admissible bit rate of pRNG-a ($C_{\min_a}$) [11].

lower design complexity as compared to oKG-D. Thus, they can be utilized for applications, which support a limited system complexity only and accept the provided level of security, i.e., CSC.

5.6 Performance Evaluation

Let us now investigate the system parameters required for the design of perfectly balanced oKG-D within oSC, when $\ell = 0$.

Fig. 5.4 numerically shows the minimal admissible bit rate of pRNG-a for oLFSR de-linearization as a function of oLFSR length n and the line rate C. The results were obtained from Eq. (5.10). As expected, the minimal rate for pRNG-a, $C_{\min_a}$, decreases with increasing oLFSR length n from hundreds of Kbits per second up

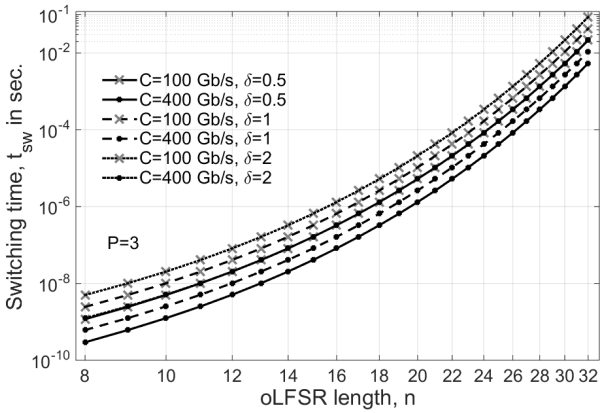


Figure 5.5: A maximal admissible switching time (t_{sw}) [11].

to few bits per second, e.g., for line rate $C = 100$ Gb/s from ≈ 400 Mb/s to ≈ 1 b/s for $n = 8$ and $n = 27$, respectively.

Let us now illustrate some basic insights of switching time t_{sw} numerically obtained from Eq. (5.8) and shown in Fig. 5.5. Here, required t_{sw} of optical switch is presented for some typical values of n and C. Since the number of parallel oLFSRs P does not significantly impact t_{sw}, i.e., $t_{\text{sw}}(P = 3) \approx t_{\text{sw}}(P = 15)$, we set $P = 3$. A careful examination of Fig. 5.5 exposes that for fixed C, t_{sw} increases with increasing n and δ. However, the increasing line rate C requires shorter switching time, e.g., if $n = 16$ and $\delta = 1$ then $t_{\text{sw}} \approx 655ns$ and $t_{\text{sw}} \approx 164ns$ for $C = 100$ Gb/s and $C = 400$ Gb/s, respectively.

With numerical results in Fig. 5.5, the required bit rate of pRNG-b C_{R_b} defined by Eq. (5.5) can be presented as a function of a maximal switching time t_{sw}; as shown in Fig. 5.6. The switching time t_{sw} was fixed for $n = 16$ using Fig. 5.5, whereby $C_{R_b} = C$, if t_{sw} is maximal possible for a certain value of n. Generally, C_{R_b} decreases with decreasing t_{sw}, C and increasing δ. In contrast, C_{R_b} is almost constant if $t_{\text{sw}} < 50$ ns, i.e., ≈ 14 Mb/s and ≈ 55 Mb/s for

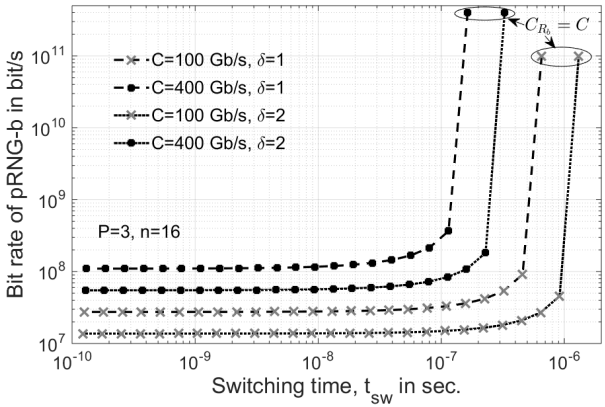


Figure 5.6: A minimal admissible bit rate of pRNG-b (C_{R_b}) [11].

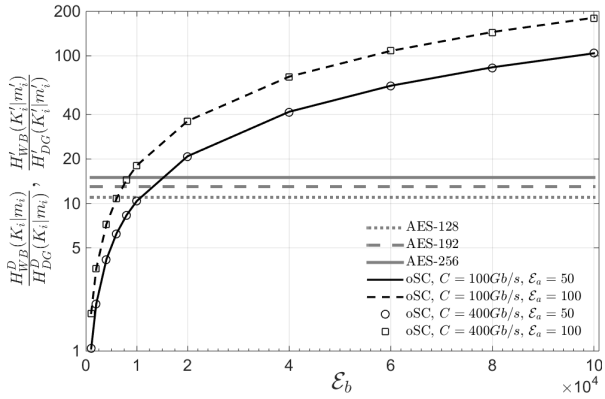


Figure 5.7: The normalized equivocation of AES and oSC [11].

$C = 100$ Gb/s and $C = 400$ Gb/s, respectively, for $\delta = 2$.

The numerical scale of physical parameters that describe the components of oKG-D is presented in Figs. 5.4, 5.5, 5.6 and shows the suitability of proposed oSC for the realization as PBC. For instance, if $C = 100$ Gb/s and $P = 3$ oLFSRs of length $n = 16$ utilized then $C_{\min_a} = 1.5$ Mb/s, $C_{\min_b} = 53.8$ Mb/s for $\delta = 1$, $t_{\text{sw}} = 321$ ns,

$\mathcal{E}_a = 50$ and $\mathcal{E}_b = 1000$. Thus, next, the computational complexity of a guessing attack on oKG-D is evaluated numerically.

Fig. 5.7 compare the key equivocation of oSC based on oKG-D and AES when an attacker uses the knowledge about system design and encryption algorithm, i.e., performs WB attack. As shown, the oSC can provide much higher equivocation than the AES algorithm independent of line rate C, if, for instance, $\mathcal{E}_a = 50$ and $\mathcal{E}_b \geq 15000$ resulting in $C_{\max_a} \approx 1.5$ Mb/s and $C_{\max_b} \approx 53.815$ Mb/s. Moreover, the guessing attack on oSC has computational complexity, which is 1 to ~ 180 times larger, in case of WB attack as compared to DG attack (only maximal 15 times larger for AES-256, i.e., $L_{k'} = 256$ bits). Generally, the computational complexity $\frac{H^{\mathrm{D}}_{WB}(K_i|m_i)}{H^{\mathrm{D}}_{DG}(K_i|m_i)}$ increases with increasing $\mathcal{E}_b$, $\mathcal{E}_a$ and independent of C, if switching time t_{sw} and related bit rate of pRNG-b are set based on Fig.5.6, i.e., Eq. (5.5).

5.7 Summary

This chapter studied the feasibility of all-optical layered encryption and, thus, the practicality of an anonymous optical network. As a result, some designs of optical stream cipher based on a cascade of standard low bit rate electronic pRNGs and oLFSR were proposed and analyzed. To increase the entropy of a generated optical key, several methods for oLFSRs *de-linearization* were investigated. A novel information-theoretical approach for security evaluation and benchmarking allowed a security assessment of the proposed optical stream cipher and its comparison with the state of the art cipher, i.e., AES. A *computationally secure cascade* and a *perfectly balanced cascade* were analytically defined as a condition for high computational security required in today's cascaded cryptosystems. The numerical evaluation of the security level provided by optical stream cipher showed that it is comparable to today's standards, such as AES while operating all-optically at high optical line-rates.

5.8 List of Symbols

Parameters unknown to attacker

m	input secret message of length L_m
k, k_a, k_b	control parameters of length L_k, L_{k_a}, L_{k_b} utilized for (re-)initialization of oKG, pRNG-a, pRNG-b, respectively.
K	optical key generated of length $L_K = L_m$
M	nonlinear bit sequence of length L_M generated by pRNG-a,b in oKG-A or oKG-B.
M_a, M_b	nonlinear bit sequence of length L_{M_a}, L_{M_b} generated by pRNG-a, pRNG-b in oKG-C,D, respectively.
$\tilde{n}$	nonlinear bit sequence of length n for oLFSR seeding
$\tilde{z}$	nonlinear bit sequence of length z for oLFSR de-linearization
α	bit sequence generated by oLFSR between two nonlinear shifts by bits from pRNG.
α_a, α_b	bit sequence generated by oLFSR between two nonlinear shifts by bits from pRNG-a and pRNG-b, respectively.
$\tilde{p}$	binary index of oLFSR randomly selected for optical key generation in oKG-B,C,D and generated by pRNG-b.

e_a, e_b	random bit sequence for definition of session bit rate of pRNG-a, pRNG-b in oKG-D, respectively.
$C_{R_a}(e_a)$, $C_{R_b}(e_b)$	randomly selected session bit rate of pRNG-a, pRNG-b in oKG-D out of $\mathcal{E}_a$, $\mathcal{E}_b$, respectively.

Parameters known to attacker

ℓ	number of bits from secret message m known to attacker
C	optical line rate
L_m	length of secret message m, m' encrypted
L_k, L_{k_a}, L_{k_b}	length of control parameters k, k_a, k_b utilized for initialization of oKG, pRNG-a, pRNG-b, respectively.
L_K	length of optical encryption key K.
L_M	length of nonlinear bit sequence M generated by pRNG-a,b in oKG-A or oKG-B.
L_{M_a}, L_{M_b}	length of nonlinear bit sequence M_a, M_b generated by pRNG-a, pRNG-b in oKG-C,D, respectively.
L_α	length of bit sequence α generated by oLFSR between two nonlinear shifts by M bits from pRNG.
L_{α_a}, L_{α_b}	length of bit sequence α_a, α_b generated by oLFSR between two nonlinear shifts, respectively.
P	number of parallel oLFSRs in oKG-B,C,D.

n	oLFSR length.
C_{R_a}, C_{R_b}	bit rate of pRNG-a, pRNG-b in oKG-A,B,C, respectively.
t_c	duration of one reset cycle.
t_{sw}	time, during which optical switch within oKG-B,C,D is able to switch between oLFSRs.
n	number of nonlinear bits generated by pRNG for oLFSR seeding
z	number of nonlinear bits generated by pRNG-a for oLFSR de-linearization.
$\mathcal{I}$	number of reset cycles required to generate an encryption key K of length L_m.
$\mathcal{E}_a$, $\mathcal{E}_b$	number of different bit rates provided by pRNG-a, pRNG-b in oKG-D, respectively.
C_{min_a}, C_{min_b}	minimum required session bit rate of pRNG-a, pRNG-b in oKG-D, respectively.
C_{max_a}, C_{max_b}	maximal possible bit rate of pRNG-a, pRNG-b defined by current technology, respectively.
L_{e_b}	length of random bit sequence e_b from control plane to define session bit rate $C_{R_b}(e_b)$ of pRNG-b in oKG-D.

6
Conclusion

This thesis studied advanced network systems and showed a great promise of parallel transmission combined with coding, random routing and redundant configurations, which is indeed the system with the maximum randomness, diversity and over-provisioning required to ensure privacy, anonymity, integrity and reliability. Even The Onion Routing (Tor) network could benefit from the advanced network concepts presented in this thesis, whereby the end-to-end performance could be improved, anonymity and privacy - increased and the censorship success - significantly reduced.

Due to some complex configurations of novel network systems designed in this thesis, the further focus was the development of advanced analysis for the evaluation of i) network performance parameters such as skew, deskew buffer size, flow completion time and information flow, ii) service reliability such as transmission reliability and Service Function Chain (SFC) reliability, and iii) network security level in case of eavesdropping and jamming. The generalized analytical models developed in the course of this thesis are based on combinatorics, probability and information theory and can be applied to any network system with intrinsic parallelism and randomness. The main findings and open issues are summarized below.

With novel performance analysis, the parallel transmission concepts with optimal and random routing were compared regarding different performance parameters. The key outcome is thereby that the parallel transmission with random routing and erasure coding can have similar performance as conventional systems over optimal paths. Random routing was shown to be feasible in combination with coding, whereby the coding and path redundancy help to reduce the skew and deskew buffer size while eliminating the need for complex path optimizations. The main advantages of these novel network systems, however, are the relatively small deskew buffer and no need for traffic re-transmission, i.e., high throughput, in the case of traffic losses or path failures. Random circuit switching is more relevant from a practical point of view than random packet switching as it requires a smaller deskew buffer at the receiver and does not lead to congestion of forwarding nodes in the network.

Applying novel SFC failure protection concepts based on systematic erasure coding or its combination with conventional backup protection, SFC reliability was extensively studied in this thesis. Here, the parallel transmission, coding, and redundancy of coding, paths and SFC were utilized. The most relevant finding was that the traffic and SFC parallelism significantly increase SFC reliability. Furthermore, a combination of coding and backup protection can additionally increase service reliability and reduce the control and management overhead related to VNF migration and traffic redirection. The novel generalized reliability analysis presented in this thesis is particularly valuable because it takes into consideration the failures of DCs, racks, servers, and VNFs as well as component sharing, heterogeneity, and their interdependency, i.e., failure propagation. Hence, a placement independent and a placement dependent SFC reliability could be analytically evaluated. This thesis provides some relevant trade-offs between heterogeneity and disjointedness of network components. The most interesting conclusion is that a highly disjoint

placement does not automatically lead to the highest SFC reliability but is a function of components' interdependency. Additionally, a possible design of a reliable parallel transmission was presented on an example of Free Space Optics. The significant result is thereby that, with a trade-off between information rate, complexity, and cost, an adaptive erasure coding controlled by an intelligent algorithm can allow highly reliable transmission independent of any interference.

Security issues such as anonymity, privacy, integrity and censorship were investigated in networks designed by combining different concepts such as serial and parallel transmission with and without coding and with optimal and random routing. The impact of parallel transmission with random routing and coding on the resulting security level was analyzed using a new generalized security analysis. That analysis assesses the privacy level and probabilities of deanonymization and censorship success when eavesdropping and jamming are performed simultaneously on different physical links and nodes. The key result is that random packet switching provides a high security degree than random circuit switching when multiple physical links are compromised. In contrast, random circuit switching showed better security performance in the networks with single wiretap edges. In circuit-switched networks, the parallel transmission with coding and random routing can significantly increase the robustness of the system against both eavesdropping and jamming as compared to a conventional optimal routing. The most important outcome, however, is the existing trade-off between the amount of coding and path redundancy and the probability of eavesdropping and jamming success. It is shown that the probability of both attacks could be minimized with a properly selected amount of coding and path redundancy. In contrast, the traditional multipath could not ensure security and reliability as a whole sent data could be eavesdropped or jammed with only one compromised physical link. This thesis additionally showed that traditional Tor can benefit from

parallel transmission implemented with multiple IP tunnels, eraser coding, redundancy and a full-random selection of onion routers and tunnels. Hence, privacy and, in case of censorship, integrity and reliability were significantly improved, whereby the anonymity was similar to the anonymity provided by traditional Tor.

In the course of this thesis, the feasibility of a novel anonymous optical network and all-optical layered encryption were investigated. Different designs of optical stream ciphers and, in particular, optical key generators were presented and analyzed. The main idea was to combine standard low bit rate pRNGs and oLFSR at line rate into cryptographic cascade to enable encryption at the line rate. A new information-theoretical approach was developed and utilized for security evaluation and benchmarking of the optical cascade. New types of cryptosystems, i.e., *computationally secure cascade* and *perfectly balanced cascade*, were introduced and analytically defined as a condition for a security level required in today's cryptosystems based on the cascade. The major result of this study is that the security provided by optical stream cipher is comparable to today's standards, e.g., AES, while operating all-optically at higher line-rates.

Although this thesis showed the suitability of the advanced network systems to improve reliability and security, some open issues need to be addressed to increase their feasibility. First, there is a need to provide a better understanding of networks with a heterogeneous level of parallelism between nodes, dynamically changing path blocking and cross-layer design in advanced anonymous networks. The parallel transmission and coding for SFC reliability result in the overhead related to synchronization of VNF replicas and processing of coded packets, which requires further investigation. Finally, the main issues related to the implementation of advanced anonymous optical networks are the optical components, which exist today only in an experimental realm and remain hard to deploy.

7
Appendix

7.1 Derivation of Eq. (2.22)

The delays of the N paths $\mathcal{P}_l \in G_{\mathcal{P}}$ are sorted in the ascending order, i.e., $d_1 \leq ... \leq d_m \leq ... \leq d_l ... \leq d_N$. In the same way, a subset of n paths $\mathcal{P}_{l_m}(\alpha) \in \mathcal{M}_n(\alpha)$ have delay values $d_{l_1}(\alpha) \leq ... \leq d_{l_m}(\alpha) \leq ... \leq d_{l_n}(\alpha)$. Due to this ordering, it is obviously that the delay of the m^{th} path $\mathcal{P}_{l_m}(\alpha) \in \mathcal{M}_n(\alpha)$ can only be mapped to delays from $\vec{d}$ in the range $d_m \leq d_{l_m}(\alpha) \leq d_{N-n+m}$. Now let us assume, that the m^{th} path $\mathcal{P}_{l_m}(\alpha)$ of an arbitrary subset $\mathcal{M}_n(\alpha)$ is mapped to the l^{th} path $\mathcal{P}_l \in G_{\mathcal{P}}$ with delay d_l. Then there are $m-1$ paths with delays $d_{l_1}(\alpha) \leq d_{l_2}(\alpha) \leq ... \leq d_{l_{m-1}}(\alpha)$ which are smaller or equal to $d_{l_m}(\alpha)$ and can be mapped to a set of $l-1$ paths with delays $d_1 \leq d_2 \leq ... \leq d_{l-1}$. Since $l \geq m$ there are $C_{l-1,m-1}$ possible combinations for this mapping. Similarly, there are $n-m$ paths with delays $d_{l_{m+1}}(\alpha) \leq ... \leq d_{l_n}(\alpha)$ larger or equal to $d_{l_m}(\alpha)$, which can be mapped to a set of $N-l$ paths with delays $d_{l+1} \leq ... \leq d_N$. Since $l \leq N-n+m$ there are $C_{N-l,n-m}$ possible combinations for this mapping. Finally, we have in total $C_{l-1,m-1}C_{N-l,n-m}$ combinations with equal occurrence probability $p_l(m)$, where the l^{th} path $\mathcal{P}_l$ with delay d_l from $\vec{d}$ is selected as m^{th} path with delay $d_{l_m}(\alpha)$. Thus, this probability can be derived as follows

$$p_l(m) = 1/a_{\mathcal{M}} \sum_{\alpha} \delta_{\alpha}(l,m) = C_{l-1,m-1}C_{N-l,n-m}/a_{\mathcal{M}}.$$

7.2 Proof of Lemma 2.1

To ensure a fault-tolerant transmission, a correction of all erroneous bits corrupted during transmission is required. Without LNC, the data blocks are only protected with the FEC code, as previously discussed. Thus, the allowed number of corrupted bits in any data block of size $\frac{L_M}{R}$ has to be at least equal to the number of bits, which can be corrected by the FEC, i.e., t_k, resulting in

$$\frac{L_M \cdot p'_e}{R} \leq t_k = \frac{h_{min} - 2}{2} = \frac{L_M - RL_M - 2R}{2R}. \tag{7.1}$$

Then, the required code rate R for fault-tolerant parallel transmission without LNC can be expressed as follows

$$R \leq \frac{L_M(1 - 2p'_e)}{L_M + 2} < 1 - 2p'_e \Leftrightarrow \forall p'_e : p'_e < 0.5. \tag{7.2}$$

Thus, that system can tolerate a bit error rate on parallel transmission paths, which is less than 50%. On the other hand, the parallel transmission system with LNC needs to detect all corrupted bits in each data block while at most half of these erroneous bits can be corrected by FEC code with capability defined as $t_k = \frac{t_e - 1}{2}$. Thus, the number of corrupted bits in LNC encoded data block has to be at most equal to t_e, i.e.,

$$\frac{L_M \cdot p'_e}{R_{LNC}} \leq t_e = h_{min} - 1 = \frac{L_M}{R_{LNC}} - L_M - 1. \tag{7.3}$$

As a result, the required code rate R_{LNC} in the system with LNC is

$$R_{LNC} \leq \frac{L_M(1 - p'_e)}{L_M + 1} < 1 - p'_e, \forall p'_e : p'_e < 1. \tag{7.4}$$

Thus, the LNC coded parallel transmission ensures fault tolerance up to 100%. Moreover, comparing code rates of the systems with and without LNC, we can observe that LNC relaxes the requirements on FEC and increases the code rate, i.e., $R_{LNC} > R$.

7.3 Proof of Lemma 2.2

Let us, first, define the information flow for both systems, with and without LNC. Using Eqs. (7.2) and (2.45), the information flow of a system without LNC can be determined as follows

$$\Phi = R < 1 - 2p'_e. \tag{7.5}$$

For the parallel transmission system with LNC, let us assume the stronger requirements on APL derived from Eq. (2.43) as $APL \geq \frac{r}{k}$, where only r out of k data blocks are allowed to be erroneous. Then, with Eqs. (7.4) and (2.45), the information flow for the parallel transmission system with LNC is defined as follows

$$\Phi_{LNC} = \frac{kR_{LNC}}{n} = \frac{kR_{LNC}}{k + k \cdot PER} = \frac{R_{LNC}}{1 + PER}. \tag{7.6}$$

With Eq. (2.41), this equation can be modified as follows

$$\Phi_{LNC} = \frac{1 - p'_e}{1 + 1 - (1 - BER)^{L_M}}. \tag{7.7}$$

The BER can be determined using Eqs. (2.42) and (7.4) as

$$\begin{aligned} BER &= \frac{L_M \cdot p'_e - R_{LNC} \cdot t_k}{L_M} \\ &= \frac{2L_M \cdot p'_e - L_M + L_M(1 - p'_e) + 2(1 - p'_e)}{2L_M} \\ &= \frac{L_M \cdot p'_e + 2(1 - p'_e)}{2L_M} = \frac{p'_e}{2} + \frac{1 - p'_e}{L_M} \approx \frac{p'_e}{2}, \end{aligned} \tag{7.8}$$

where L_M is a large number, e.g., $L_M \geq 64$ bits in the Ethernet layer. Thus, the information flow of the proposed system is

$$\Phi_{LNC} = \frac{1 - p'_e}{2 - (1 - \frac{p'_e}{2})^{L_M}} \approx \frac{1 - p'_e}{2}, \forall p'_e > 0. \tag{7.9}$$

The resulting functions of an information flow are defined by Eqs. (7.5) and (7.9) for systems without and with LNC, respectively. Generally, the information flows in both systems decrease with increasing

bit error rate p'_e, while these functions (Eqs. (7.5) and (7.9)) have one joint value only, which can be determined with $\Phi_{LNC} = \Phi$ as

$$\frac{1 - p'_e}{2} = 1 - 2p'_e, \tag{7.10}$$

$$p'_e = \frac{1}{3}. \tag{7.11}$$

The overhead (Eq.(2.46)) for both defined transmission systems increases with growing p'_e, whereby $\Theta = 1 - 1 + 2p'_e = 2p'_e$ and $\Theta_{LNC} = 1 - \frac{1-p'_e}{2} = \frac{1+p'_e}{2}$ have a joint value, when $p'_e = \frac{1}{3}$. As a result, $\Theta_{LNC} < \Theta$, i.e., $\frac{1+p'_e}{2} < 2p'_e$, if $p'_e > \frac{1}{3}$.

7.4 Derivation of Eq. (2.47)

The routing over the network is implemented over k paths and r redundant paths in parallel. Thus, $n = k + r$ of N existing paths can be chosen randomly. For decoding to start, the receiver needs only k data blocks from any of those k paths that are shorter than other r paths. Since the vector $\vec{d}$ is sorted in ascending order as $d_1 \leq d_2 \leq ... \leq d_k$, a maximal path delay that has an impact on decoding start is d_k, and, thus, the delay d_k from $\vec{d}$ defines a minimal possible value of a maximal path delay. On the other hand, the last r delays from vector $\vec{d}$ can be any delay d_l, $N - r + 1 \leq l \leq N$. Since only paths with delays d_{N-k-r}, $d_{N-k-r+1}$, ..., $d_{N-k-r+k} = d_{N-r}$ provide data blocks relevant for decoding to start, the path delay d_{N-r} is a maximum possible delay at the receiver.

7.5 Proof of Lemma 3.1

When a parallelized SFC consists of $\Psi = 1$ VNF types and all n VNFs have the same VNF reliability $p_4 < 1$, i.e., there is only one reliability class ($N = 1$), the SFC reliability calculation follows the parallel reliability model described by Binomial formula. Then, the

service success is defined as a probability that at least k out of $\Lambda_4 = n$ components are available, i.e., $R(N=1) = \sum_{f_4=0}^{A_4} p(\Lambda_4, f_4, p_4)$.

When, for instance, the reliability of all active VNFs and all backup VNFs of a certain type is p_{4_1} and p_{4_2}, respectively, and $p_{4_1} \neq p_{4_2}$, there are two reliability classes, $N = 2$. Then, the SFC reliability is determined as a probability that at least k over all $\Lambda_{4_1} + \Lambda_{4_2} = n_1 + n_2 = k + r = n$ VNFs do not fail and at least k sub-SFCs can be composed to serve k sub-flows. Since we assumed SFC of one VNF, $\Psi = 1$, at least $k - f_{4_1}$ active VNFs, where $0 \leq f_{4_1} \leq min\{k, r\}$, and at least f_{4_1} backup VNFs have to be available, while also f_{4_2} backup VNFs can fail without service interruption, where $0 \leq f_{4_2} \leq r - f_{4_1}$. Thus, AVF of active and backup VNFs is determined as $A_{4_1} = min\{k, r\}$ and $A_{4_2} = min\{r, r - f_{4_1}\} = r - f_{4_1}$, respectively. Following the Binomial formula and, additionally, applying the serial reliability model as VNFs from both reliability classes, i.e., at least k, have to be available at the same time; the SFC reliability is determined as $R_4(N = 2) = \sum_{f_{4_1}=0}^{A_{4_1}} p(\Lambda_{4_1}, f_{4_1}, p_{4_1}) \sum_{f_{4_2}=0}^{A_{4_2}} p(\Lambda_{4_2}, f_{4_2}, p_{4_2})$, when $\Psi = 1$, $\Lambda_{4_1} = k$, $\Lambda_{4_2} = r$ and $N = 2$.

It is obvious that this example can be extended to $N \leq k + r$ reliability classes, whereby there is a need to extend the equation above to up to N summations. In this case, the AVF of any reliability class ξ is calculated as $A_{4_\xi} = min\{n_\xi, r - \sum_{\xi=1}^{\xi-1} f_{4_\xi}\}$ to ensure that the number of failed VNFs over all reliability classes, i.e., from 1 to $\xi - 1$, is not larger then the number of backup VNFs provided. Thus, the SFC reliability for $\Psi = 1$ and $N \geq 1$ can be generalized as

$$R'(N) = \prod_{\xi=1}^{N} \sum_{f_{4_\xi}=0}^{A_{4_\xi}} p(\Lambda_{4_\xi}, f_{4_\xi}, p_{4_\xi}) = \sum_{f_{4_1}=0}^{A_{4_1}} p(\Lambda_{4_1}, f_{4_1}, p_{4_1}) \ldots \sum_{f_{4_N}=0}^{A_{4_N}} p(\Lambda_{4_N}, f_{4_N}, p_{4_N}), \tag{7.12}$$

where $R'(N)$ is a probability that there are at least k available out of n VNFs of a certain type, e.g., VNF1, over all N reliability classes.

Next, let us assume that the parallelized SFC consists of $\Psi > 1$ different VNFs. Then, the SFC reliability is determined as a probability that at least k over all $k + r = n$ VNFs of each type do not fail, and at least k sub-SFCs can be composed by $k\Psi$ VNFs of each type to serve k sub-flows. When all VNFs of the same type have the same VNF reliability, e.g., all VNF1 and all VNF2 have reliability p_{4^1} and p_{4^2}, respectively, the SFC reliability is a probability that at least k VNFs of each type, e.g., k VNF1 and k VNF2, are available as described by $R(N = 1)$ above. Applying the serial reliability model, the SFC reliability is $R^{1...\Psi} = R^1(N = 1) \cdot R^2(N = 1)...R^\Psi(N = 1) = \prod_{\psi=1}^{\Psi} R^\psi(N = 1) = [R(N = 1)]^\Psi$. In case all VNFs of any type, i.e., from VNF1 to VNFΨ, utilized to build a certain SFC of reliability class ξ have the same VNF reliability value, i.e., $p_{4^1_\xi} = p_{4^2_\xi} = ... = p_{4^\Psi_\xi} = p_{4_\xi}$, at most $A_4 = r$ VNFs of any type can fail without any impact on service maintenance. Then, the SFC reliability can be generalized following the serial reliability model for the whole SFC of Ψ VNFs as follows

$$R = [R(N = 1)]^\Psi = \left[\sum_{f_4=0}^{A_4} p(\Lambda_4, f_4, p_4) \right]^\Psi . \tag{7.13}$$

Similarly, when there are N reliability classes and any reliability class ξ contains n_ξ sub-SFCs, all $k_\xi\Psi$ VNFs from the same reliability class have the same VNF reliability p_{4_ξ}, the overall SFC reliability can be derived from Eq. (7.12) as $R(N, \Psi) = R'^{1...\Psi}(N) = \prod_{\psi=1}^{\Psi} R'^\psi(N) = [R'(N)]^\Psi$, which results in Eq. (3.19).

7.6 Proof of Lemma 3.2

Let us first assume that SFC consists of $\Psi = 1$ VNF types, e.g., VNF1 only, while all $n = k + r$ active and backup VNFs have the

same VNF components reliability p_4 as well as all DCs, racks and servers involved in placement of these n VNFs have the same component reliability p_1, p_2 and p_3, respectively. Then, the probability of service success can be calculated with Binomial formula and the serial reliability model as previously discussed in Sec. 3.4.6, whereby it is required to consider the availability of all component types involved $1 \leq c \leq \mathcal{C}$, i.e., a minimal number of DC, racks and servers is required to ensure availability of at least k out of n VNFs, i.e.

$$R(\Psi = 1) = \prod_{c=1}^{\mathcal{C}} \sum_{f_c=0}^{A_c} p(\Lambda_c, f_c, p_c) = \sum_{f_1=0}^{A_1} p(\Lambda_1, f_1, p_1) \cdot \sum_{f_2=0}^{A_2} p(\Lambda_2, f_2, p_2) \cdot \sum_{f_3=0}^{A_3} p(\Lambda_3, f_3, p_3) \cdot \sum_{f_4=0}^{A_4} p(\Lambda_4, f_4, p_4). \tag{7.14}$$

To extend Eq. (7.14) to arbitrary SFC length, i.e., $\Psi \geq 1$ VNFs, there is a need to consider the VNF placement strategy and the resulting level of disjointedness Δ. Since our placement strategies are based on SFC or VNF type, $\mathcal{C} - \Delta$ component types are the shared components and host a whole sub-SFC. Thus, the SFC reliability calculation depends on the level of disjointedness, i.e., Δ. Then, the serial reliability model can be only applied to $R(\Psi = 1)$, similarly as for Eq. (7.13), when each VNF type of a certain sub-SFC is allocated to different components of any hierarchy level c, $1 \leq c \leq \mathcal{C}$, resulting in $\Delta = 4$. In this case, any VNF type is placed in different DC, different racks, different servers and VMs and the SFC reliability is determined with Eq. (7.14) as $R_{\Delta=4} = \prod_{\psi=1}^{\Psi} R(\Psi = 1) = [R(\Psi = 1)]^{\Psi} = \left[\sum_{f_1=0}^{A_1} P(\Lambda_1) \sum_{f_2=0}^{A_2} P(\Lambda_2) \sum_{f_3=0}^{A_3} P(\Lambda_3) \sum_{f_4=0}^{A_4} P(\Lambda_4) \right]^{\Psi}$. When different VNF types, i.e., sub-SFCs, are placed in the same DC but different racks, servers and VMs ($\Delta = 3$), the failure of DC, i.e., the shared component, affects all Ψ VNFs of a certain sub-SFC and, thus, needs to be considered once and not separately for each out of Ψ

VNFs. Thus, the SFC reliability can be calculated by separation the probability that there are at least $\Lambda_1 - A_1$ available DCs required for maintenance of at least k sub-SFCs, while the serial reliability model is still valid: $R_{\Delta=3} = \sum_{f_1=0}^{A_1} P(\Lambda_1) \left[\sum_{f_2=0}^{A_2} P(\Lambda_2) \sum_{f_3=0}^{A_3} P(\Lambda_3) \sum_{f_4=0}^{A_4} P(\Lambda_4) \right]^{\Psi}$. Similarly, when different VNF types, i.e., sub-SFCs, are placed in the same DC, same rack and separate servers resulting in disjointedness level $\Delta = 2$, there are 2 shared components, i.e., DC and rack. Thus, any failure of DC or rack will affect all Ψ VNFs and needs to be considered once for calculating the probability that at least $\Lambda_1 - A_1$ DCs and $\Lambda_2 - A_2$ racks are available to build k sub-SFCs of Ψ VNFs each. Then, the SFC reliability is determined as $R_{\Delta=2} = \sum_{f_1=0}^{A_1} P(\Lambda_1) \sum_{f_2=0}^{A_2} P(\Lambda_2) \left[\sum_{f_3=0}^{A_3} P(\Lambda_3) \sum_{f_4=0}^{A_4} P(\Lambda_4) \right]^{\Psi}$. A VNF disjointedness only ($\Delta = 1$) is possible when different VNF types are placed in the same server and, thus, in the same rack and same DC, resulting in three shared component types. Consequently, a failure of the server, rack or DC will lead to a failure of a whole sub-SFC, i.e., of all Ψ VNFs. Thus, only VNFs itself are independent of each other resulting in SFC reliability defined as $R_{\Delta=1} = \sum_{f_1=0}^{A_1} P(\Lambda_1) \sum_{f_2=0}^{A_2} P(\Lambda_2) \cdot \sum_{f_3=0}^{A_3} P(\Lambda_3) \cdot \left[\sum_{f_4=0}^{A_4} P(\Lambda_4) \right]^{\Psi}$. This means that the number of summations outside and inside the brackets, in the formulas for reliability calculation above, is a function of Δ and a number of shared components. Thus, the summations for the shared components are outside the brackets, i.e., $\mathcal{C} - \Delta$ summations, and the Δ summations for component types whose failure impacts one VNF type only, are placed inside the brackets. Without loss of generality, the probability of service success for any value Δ is determined by Eq. (3.20).

7.7 Proof of Lemma 3.3

First, we assume that the components from different reliability classes do not have any common root components, which corresponds to the Inter-DCN placement of active and backup sub-SFCs. Thus, the SFCs of a certain reliability class ξ are placed disjointedly from any other reliability classes. Assuming that any SFC contains only one VNF ($\Psi = 1$), additionally relaxes the placement dependency as there can not be any shared components. Then, the SFC reliability can be determined based on Eqs. (7.12) and (7.14) as follows

$$R(\Psi = 1, N) = \prod_{\xi=1}^{N} R(\Psi = 1) = \prod_{\xi=1}^{N} \prod_{c_\xi=1}^{\mathcal{C}} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} p(\Lambda_{c_\xi}, f_{c_\xi}, p_{c_\xi}), \quad (7.15)$$

where failure and availability of each component type and each reliability class are described by their own summation. Thus, all $\mathcal{C}$ component types, i.e., from type 1 to type $\mathcal{C}$ are considered for each reliability class ξ, $1 \leq \xi \leq N$. Thus, any summation determines the probability that service will not fail in case of f_{c_ξ} failures of any component type c of a reliability class ξ, whereby at most A_{c_ξ} failures of this component type c and class ξ are allowed.

Let us next assume that some sub-SFCs from different reliability classes have common root components, and each sub-SFC consists of one VNF ($\Psi = 1$) and, thus, there are no shared components. The failure of the common root components results in the failure of entire sub-SFCs and related component types from different reliability classes. All reliability classes combined by the common roots are collected in a set Φ. Any failure of any component type c_{w_ρ} related to the reliability classes from a set w_ρ affects ACF A_{c_ξ} and the number of available components Λ_{c_ξ} of all related classes ξ, i.e., $\xi \in w_\rho$. Then, to take into account the common roots of different

reliability classes, Eq. (7.15) for $\Psi = 1$ is modified as follows

$$R(\Psi = 1, N, \Phi) = \prod_{\rho=1}^{|\Phi|} \sum_{f_{c_{w_\rho}}=0}^{A_{c_{w_\rho}}} P(\Lambda_{c_{w_\rho}}) \prod_{\xi=1}^{N} \prod_{\substack{c_\xi=1 \\ \phi(c_\xi)=1}}^{\mathcal{C}} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi}), \quad (7.16)$$

where $\Lambda_{c_{w_\rho}} = 1 = const$ as there can be only one common root c_{w_ρ} for reliability classes from set w_ρ. The function $\phi(c_\xi)$ determined by Eq. (3.22) ensures that the common root components for any reliability class $\xi \in w_\rho$ are considered only once, i.e., by the first summation over $f_{c_{w_\rho}}$.

Next, let us assume that SFC consists of $\Psi \geq 1$ VNFs and there are no common root components, but some sub-SFCs utilize shared components to place Ψ different VNFs. As any reliability class, ξ, has a level of disjointedness Δ_ξ, where $1 \leq \Delta_\xi \leq \mathcal{C}$, the expression for the SFC reliability needs to consider the level of disjointedness Δ_ξ as well. Only considering the reliability class ξ which provides the level of disjointedness Δ_ξ due to a certain placement strategy, there are $\mathcal{C} - \Delta_\xi$ shared components, i.e., shared by different VNF types, i.e., components $c_\xi = 1_\xi, ..., (\mathcal{C} - \Delta_\xi)_\xi$, and Δ_ξ disjoint components, i.e., $c_\xi = (\mathcal{C} - \Delta_\xi + 1)_\xi, ..., \mathcal{C}_\xi$. Then, similar to Eq. (3.20), there are $\mathcal{C}_\xi$ summations for each reliability class ξ, where all summations for shared components and all summations for disjoint components should be separated. That is because each summation for disjoint components is valid for each out of Ψ VNF types. Thus, the SFC reliability is determined as follows

$$R_{\Delta_1,...,\Delta_N} = \prod_{\xi=1}^{N} \prod_{c_\xi=1}^{\mathcal{C}_\xi - \Delta_\xi} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi}) \left[\prod_{\xi=1}^{N} \prod_{c_\xi=\mathcal{C}_\xi - \Delta_\xi + 1}^{\mathcal{C}_\xi} \sum_{f_{c_\xi}=0}^{A_{c_\xi}} P(\Lambda_{c_\xi}) \right]^{\Psi}, \quad (7.17)$$

where any A_{c_ξ} and A'_{c_ξ} describes ACF of a shared and disjoint component type c_ξ of any reliability class ξ, respectively. When a level of disjointedness is maximal, i.e., $\Delta_\xi = \mathcal{C}$, all component types are

disjoint and their number is the same for any out of Ψ VNFs, i.e., all $\mathcal{C}$ related summations are inside the brackets in Eq. (7.17), i.e., $\prod_{c=1}^{\mathcal{C}-\Delta}\sum_{f_c=0}^{A_c}P(\Lambda_c)=1$, if $\Delta=\mathcal{C}$. Similar, in the case of $\Delta_\xi=1$, one component type, i.e., $c=\mathcal{C}$, is disjoint, resulting in one summation per reliability class ξ inside brackets. When $\Delta_\xi=\mathcal{C}-1$, i.e., only component type $c=1$, i.e., DCs, is a shared component of all Ψ different VNF types of reliability class ξ, there is one summation outside the brackets. That summation describes the probability that there are enough available active or backup components of type $c=1$ to maintain the service.

When there are some shared components for VNFs from sub-SFCs of length $\Psi\geq 1$ as well as common roots for different reliability classes described by a set Φ, we can derive the SFC reliability with Eqs. (7.17) and (7.16). The resulting equation for the service success takes into account $N\geq 1$ reliability classes of SFCs of length $\Psi\geq 1$, their common root and shared components, and failures of any out of $\mathcal{C}$ component type involved in placement of a sub-SFCs from a certain reliability class ξ, i.e., $R_{\Delta_1,\ldots,\Delta_N}(\Phi)=\prod_{\rho=1}^{|\Phi|}\sum_{f_{c_{w_\rho}}=0}^{A_{c_{w_\rho}}}P(\Lambda_{c_{w_\rho}})R_{\Delta_1,\ldots,\Delta_N}$ and presents Eq. (3.21).

Finally, let us derive Eq. (3.22), where $\phi(c_\xi)$ needs to ensure that the common root components are considered only once by the first summation in Eqs. (3.21) and (7.16) and only component type c_ξ, which is not the common root component, is considered in the following summation. That allows us to calculate the probability that there are enough available components of type c_ξ to maintain the service. As a result, c_ξ needs to be additionally considered, when either the reliability class ξ of considered component type c_ξ does not have a common root components with other reliability classes, i.e., $\forall\rho:\xi\notin w_\rho$, or ξ have some common root components with other reliability classes, i.e., $\xi\in w_\rho$, but the considered component type c is not the common root, i.e., $c_\xi\neq c_{w_\rho}$.

7.8 Derivation of Eq. (3.27)

As all components of a certain reliability class are placed separately from components of any other reliability class and only some common root components c_{w_ρ} can combine different reliability classes, there is a need to take into account each reliability class ξ individually. Here, Λ_{c_ξ} is a function of VNF placement strategy and a number of available and failed components related to the reliability class ξ. Since DCs are the components from the highest hierarchy level and independent of failures of any other component types, the amount of available DCs is equal to the amount of data centers required to place n_ξ sub-SFCs of reliability class ξ, i.e., $\Lambda_{1_\xi} = \bar{n}_{1_\xi}$. The failure of f_{1_ξ} DCs causes a failure of $f_{1_\xi} n_{2_\xi}$ racks reducing the overall number of available racks as $\Lambda_{2_\xi} = (\Lambda_{1_\xi} - f_{1_\xi}) n_{2_\xi}$, where $\Lambda_{2_\xi} \equiv \Lambda_{c_\xi} = (\Lambda_{c-1_\xi} - f_{c-1_\xi}) n_{c_\xi}$. Then, the remaining amount of servers after DC and rack failures is $\Lambda_{3_\xi} = ((\Lambda_{1_\xi} - f_{1_\xi}) n_{2_\xi} - f_{2_\xi}) n_{3_\xi} = (\Lambda_{2_\xi} - f_{2_\xi}) n_{3_\xi}$, where $\Lambda_{3_\xi} \equiv \Lambda_{c_\xi} = (\Lambda_{c-1_\xi} - f_{c-1_\xi}) n_{c_\xi}$. After failures of f_{1_ξ} DCs, f_{2_ξ} racks and f_{3_ξ} servers the remaining amount of VNFs of any type is $\Lambda_{4_\xi} = (((\Lambda_{1_\xi} - f_{1_\xi}) n_{2_\xi} - f_{2_\xi}) n_{3_\xi} - f_{3_\xi}) n_{4_\xi} = (\Lambda_{3_\xi} - f_{3_\xi}) n_{4_\xi} \equiv (\Lambda_{c-1_\xi} - f_{c-1_\xi}) n_{c_\xi}$, if $c_\xi = 4$. Since some reliability classes have a common root component c_{w_ρ}, which impact any reliability class ξ, $\xi \in w_\rho$, there is a need to consider the availability and failure of common root components from a set Φ as $n_{c_\xi} \equiv n_{c_{w_\rho}} = \Lambda_{c_{w_\rho}}$ for any $\xi \in w_\rho$. Thus, the general expression for the number of available components of any type c from a reliability class ξ is represented by Eq. (3.27).

7.9 Proof of Lemma 5.1

Let us consider Eqs. (5.19), (5.22) and (5.26). The condition $H^{\mathrm{D}}_{DG}(K_i|m_i) \geq H^{\mathrm{D}}_{BB}(K_i|m_i)$ is valid because $L_m - \ell \geq L_{k_a} + L_{k_b} + L_{e_b}$ per system design, if $\ell << L_m$. With Eqs. (5.27) and (5.28), the condition

$H^{\mathrm{D}}_{WB}(K_i|m_i) \geq H^{\mathrm{D}}_{DG}(K_i|m_i)$ results in

$$(n+p+z_\ell(\mathcal{E}_a,e_b))\,\frac{L_m-\ell}{C}\sum_{e_b=1}^{\mathcal{E}_b}\frac{C_{R_b}(e_b)}{\big(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\big)} \geq L_m-\ell, \tag{7.18}$$

whereby we can modify Eq. (7.18) as follows

$$\frac{(n+p)}{C}\sum_{e_b=1}^{\mathcal{E}_b}\frac{C_{R_b}(e_b)}{\big(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\big)}+ + \frac{1}{C}\sum_{e_b=1}^{\mathcal{E}_b}\frac{z_\ell(\mathcal{E}_a,e_b)C_{R_b}(e_b)}{\big(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\big)} \geq 1. \tag{7.19}$$

With Eqs. (5.28) and (5.29), we can simplify Eq. (7.19) and solve the second sum as follows

$$\frac{(n+p)}{C}\sum_{e_b=1}^{\mathcal{E}_b}\frac{C_{R_b}(e_b)}{\big(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\big)}+ + \frac{\mathcal{E}_b\mathcal{E}_a(\mathcal{E}_a-1+2C_{\min_a})}{2C} \geq 1, \tag{7.20}$$

whereby that constraint is valid if at least one of the summands is equal to 1. That can be employed with the configurable parameters $\mathcal{E}_a$ and $\mathcal{E}_b$. Eq. (7.20) can be modified to a quadratic equation as a function of $\mathcal{E}_a$, i.e.,

$$\mathcal{E}_a^2\mathcal{E}_b+\mathcal{E}_a\mathcal{E}_b(2C_{\min_a}-1)+ + 2(n+p)\sum_{e_b=1}^{\mathcal{E}_b}\frac{C_{R_b}(e_b)}{\big(p+n+t_{\mathrm{sw}}C_{R_b}(e_b)\big)}-2C \geq 0. \tag{7.21}$$

As $\mathcal{E}_a > 0$, Eq. (7.21) can be solved, while the required number of different bit rates of pRNG-a is, then, presented by Eq. (5.33).

Bibliography

[1] A. Engelmann, W. Bziuk, A. Jukan, and M. Médard, "Exploiting parallelism with random linear network coding in high-speed ethernet systems," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2829–2842, Dec 2018.

[2] "ISO/IEC/IEEE International Standard for Ethernet," *ISO/IEC/IEEE 8802-3:2014(E)*, pp. 1–3754, April 2014.

[3] A. Engelmann and A. Jukan, "Away from optimal: Performance analysis of multilane ethernet transmission with random path selection in optical networks," in *2015 International Conference on Optical Network Design and Modeling (ONDM)*, May 2015, pp. 116–121.

[4] A. Engelmann and A. Jukan, "Serial, parallel or hybrid: Towards a highly reliable transmission in rf/fso network systems," in *2015 IEEE International Conference on Communications (ICC)*, June 2015, pp. 6181–6186.

[5] A. Engelmann, W. Bziuk and A. Jukan, "Bounding reliability in service function chaining," in *IEEE International Conference on Information, Communication and Electronic Technology (MIPRO)*, 2020, pp. 413-418.

[6] A. Engelmann, A. Jukan, and R. Pries, "On coding for reliable vnf chaining in dcns," in *2019 15th International Conference*

on the Design of Reliable Communication Networks (DRCN), March 2019, pp. 83–90.

[7] A. Engelmann and A. Jukan, "A Combinatorial Reliability Analysis of Generic Service Function Chains in Data Center Networks," in *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, under submission.

[8] A. Engelmann, S. Zhao, and A. Jukan, "Improving security in optical networks with random forwarding and parallel transmission," in *IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–6.

[9] A. Engelmann and A. Jukan, "Balancing the demands of reliability and security with linear network coding in optical networks," in *IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–7.

[10] A. Engelmann and A. Jukan, "Defying censorship with multicircuit tor and linear network coding," in *12th CMI Conference on Cybersecurity and Privacy (CMI)*, Nov 2019, pp. 1–6.

[11] A. Engelmann and A. Jukan, "Toward all-optical layered encryption: A feasibility analysis of optical stream cipher," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, pp. 2689–2704, Oct 2019.

[12] W. Stallings, *Network Security Essentials: Applications and Standards*, 6th ed. Pearson, 2016.

[13] J. R. Vacca, *Network and System Security, Second Edition*, 2nd ed. Syngress Publishing, 2013.

[14] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach (7th Edition)*, 7th ed. Pearson, 2017.

[15] D. Eppstein, "Finding the k shortest paths," *SIAM J. Comput.*, vol. 28, no. 2, pp. 652–673, Feb. 1999.

[16] S. Nelakuditi and Z.-L. Zhang, *On selection of paths for multipath routing.* Springer, 2001, pp. 170–184.

[17] H. Xu and B. Li, "Tinyflow: Breaking elephants down into mice in data center networks," in *IEEE 20th International Workshop on Local Metropolitan Area Networks*, May 2014, pp. 1–6.

[18] S. Chakraborty and C. Chen, "A low-latency multipath routing without elephant flow detection for data centers," in *IEEE 17th International Conference on High Performance Switching and Routing*, 2016, pp. 49–54.

[19] A. Mallik and S. Hegde, "A novel proposal to effectively combine multipath data forwarding for data center networks with congestion control and load balancing using software-defined networking approach," in *2014 International Conference on Recent Trends in Information Technology*, April 2014, pp. 1–7.

[20] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society of Industrial and Applied Mathematics*, vol. 8, no. 2, p. 300–304, June 1960.

[21] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. on Inform. Theory*, vol. 56, no. 9, pp. 4539–4551, Sept 2010.

[22] R. Koetter and M. Médard, "An algebraic approach to network coding," in *IEEE/ACM Trans. Networking*, vol. 11, 2003, pp. 782–295.

[23] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, Oct 2006.

[24] N. Maxemchuk, "Dispersity Routing: Past and Present," in *IEEE Military Communications Conference*, 2007, pp. 1–7.

[25] W. Ahmed and Y. W. Wu, "A survey on reliability in distributed systems," *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1243 – 1255, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022000013000652

[26] H. Nakamura et al., "ETSI GS NFV-REL 003 V1.1.1: Network Functions Visualisation (NFV); Reliability; Report on Models and Features for End-to-End Reliability."

[27] A. Immonen and D. Pakkala, "A survey of methods and approaches for reliable dynamic service compositions," *Service Oriented Computing and Applications*, vol. 8, no. 2, pp. 129–158, Jun 2014.

[28] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr, "Towards an analysis of onion routing security," in *International workshop on designing privacy enhancing technologies: design issues in anonymity and unobservability*. Springer-Verlag N-Y, Inc., 2001, pp. 96–114.

[29] E. Erdin, C. Zachor, and M. H. Gunes, "How to find hidden users: A survey of attacks on anonymity networks," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2296–2316, 2015.

[30] S. Nepal, S. Dahal, and S. Shin, "Deanonymizing schemes of hidden services in tor network: A survey," in *International*

Conference on Information Networking (ICOIN), Jan 2015, pp. 468–473.

[31] R. Dingledine and N. Mathewson, "Tor protocol specification," 2018. [Online]. Available: https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt

[32] M. AlSabah, K. Bauer, T. Elahi, and I. Goldberg, "The path less travelled: Overcoming tor's bottlenecks with traffic splitting," in *Privacy Enhancing Technologies*. Springer Berlin Heidelberg, 2013, pp. 143–163.

[33] H. T. Karaoglu, M. B. Akgun, M. H. Gunes, and M. Yuksel, "Multi path considerations for anonymized routing: Challenges and opportunities," in *International Conference on New Technologies, Mobility and Security*, May 2012, pp. 1–5.

[34] F. Rochet, O. Pereira and O. Bonaventure, "Moving Tor circuits towards multiple-path : Anonymity and performance considerations," 2015.

[35] L. Yang and F. Li, "Enhancing traffic analysis resistance for Tor hidden services with multipath routing," in *IEEE Conference on Communications and Network Security*, Sep. 2015, pp. 745–746.

[36] L. Yang and F. Li, "mTor: A multipath Tor routing beyond bandwidth throttling," in *IEEE Conference on Communications and Network Security*, Sept 2015, pp. 479–487.

[37] U. M. Maurer and J. L. Massey, "Cascade ciphers: The importance of being first," *Journal of Cryptology*, vol. 6, no. 1, pp. 55–61, Mar 1993.

[38] R. Prosser, "Routing procedures in communications networks-part i: Random procedures," *IRE Transactions on Communications Systems*, vol. 10, no. 4, pp. 322–329, December 1962.

[39] J. Moughton, J. Swift, P. Baxendale, and P. Mars, "Decentralised routing strategies in sparsely connected circuit switched networks," in *IEE Eighth UK Teletraffic Symposium*, April 1991, pp. 7/1–7/5.

[40] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The Benefits of Coding Over Routing in a Randomized Setting," in *IEEE International Symposium on Information Theory*, 2003, pp. 442-.

[41] C. C. Wang and M. Chen, "Sending perishable information: Coding improves delay-constrained throughput even for single unicast," *IEEE Trans. on Inform. Theory*, vol. 63, no. 1, pp. 252–279, Jan 2017.

[42] M. Chen, Y. Tian, and C. C. Wang, "On coding capacity of delay-constrained network information flow: An algebraic approach," in *IEEE International Symposium on Information Theory (ISIT)*, July 2016, pp. 2908–2912.

[43] N. Calabretta and H. Dorren, "All-optical label processing in optical packet switched networks," in *Optical Fiber Communication (OFC)*, March 2010, pp. 1–3.

[44] A. Anand, V. K. Sihag, and S. N. Gupta, "Wavelength conversion and deflection routing in all-optical packet-switched networks through contention resolution: A survey," in *Proceedings of the CUBE International Information Technology Conference*, ser. CUBE '12, 2012.

[45] C. Lee, "Analysis of switching networks," *The Bell System Technical Journal*, vol. 34, no. 6, pp. 1287–1315, Nov 1955.

[46] L. Kleinrock, *Queueing Systems: Volume 1: Theory*. New York, NY, USA: Wiley-Interscience, 1975.

[47] O. Trullols-Cruces, J. M. Barcelo-Ordinas, and M. Fiore, "Exact decoding probability under random linear network coding," *IEEE Communications Letters*, vol. 15, no. 1, pp. 67–69, January 2011.

[48] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A Random Linear Network Coding Approach to Multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, October 2006.

[49] C. F. Chiasserini, E. Viterbo, and C. Casetti, "Decoding probability in random linear network coding with packet losses," *IEEE Communications Letters*, vol. 17, no. 11, pp. 1–4, November 2013.

[50] F. Li, "Exact decoding probability of random linear network coding for combinatorial networks," *Wuhan University Journal of Natural Sciences*, vol. 20, no. 5, pp. 391–396, 2015.

[51] H. Han, D. Peng, and L. Wang, "Upper bounds for the failure probability of random linear network coding for multicast network," in *IET networks*, vol. 2, no. 4, pp. 181–187, December 2013.

[52] K.-H. Lee, J.-H. Kim, and S. Cho, "RLNC in Practical Wireless Networks" in *9th International Conference on Wireless Algorithms, Systems, and Applications (WASA)*, Springer International Publishing, 2014, pp. 194–204.

[53] "ITU-T recommendation g. 7041/y. 1303, generic framing procedure," 2019.

[54] R. Essiambre, G. Kramer, P. Winzer, G. Foschini, and B. Goebel, "Capacity Limits of Optical Fiber Networks," *IEEE Journal of Lightwave Technology*, vol. 28, no. 4, pp. 662–701, Feb. 2010.

[55] W. Freude, R. Schmogrow, B. Nebendahl, M. Winter, A. Josten, D. Hillerkuss, S. Koenig, J. Meyer, M. Dreschmann, M. Huebner, C. Koos, J. Becker, and J. Leuthold, "Quality metrics for optical signals: Eye diagram, q-factor, osnr, evm and ber," in *2012 14th International Conference on Transparent Optical Networks (ICTON)*, July 2012, pp. 1–4.

[56] E. Vanin, "Performance Evaluation of Intensity Modulated Optical OFDM System with Digital Baseband Distortion," *Opt. Express*, vol. 19, no. 5, pp. 4280–4293, Feb. 2011.

[57] P. Vitthaladevuni, M.-S. Alouini, and J. Kieffer, "Exact BER Computation for Cross QAM Constellations," *IEEE Transactions on Wireless Communications*, vol. 4, no. 6, pp. 3039–3050, Nov 2005.

[58] J. Sun, Y. Zhang, X. Wang, S. Xiao, Z. Xu, H. Wu, X. Chen, and Y. Han, "Dc2-mtcp: Light-weight coding for efficient multi-path transmission in data center network," in *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2017, pp. 419–428.

[59] M. Tatarko, L. Ovsenik, and J. Turán, "Availability and reliability of fso links estimation from measured fog parameters," in *MIPRO, Proceedings of the 35th International Convention.* IEEE, 2012, pp. 192–195.

[60] J. Shapiro and A. Puryear, "Reciprocity-enhanced optical communication through atmospheric turbulence; part i: Reciprocity proofs and far-field power transfer optimization," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 4, no. 12, pp. 947–954, Dec 2012.

[61] A. Puryear, J. Shapiro, and R. Parenti, "Reciprocity-enhanced optical communication through atmospheric turbu-

lence; part ii: Communication architectures and performance," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 5, no. 8, pp. 888–900, Aug 2013.

[62] R. Parenti, J. Roth, J. Greco, F. Walther, and J. Shapiro, "Channel reciprocity in single-mode free-space optical links," in *IEEE Photonics Society Summer Topical Meeting Series*, July 2012, pp. 113–114.

[63] M. Khalighi and M. Uysal, "Survey on free space optical communication: A communication theory perspective,"in *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 2231–2258, 2014.

[64] J. Anguita, M. Neifeld, and B. Vasic, *Multi-beam free-space optical link using space-time coding.* Edgewood, Maryland, USA: Scitech Publishing, Dec. 2009.

[65] Y. H. Kho, L. Yong, K. L. Lau and K. P. Kiu, "Design of an indoor wireless optical transceiver system with source and channel coding," in *IEEE Symposium on Industrial Electronics and Applications*, Sept 2012, pp. 45–49.

[66] L. B. Stotts, N. Plasson, T. W. Martin, D. W. Young and J. Juarez, "Progress towards reliable free-space optical networks," in *Military Communications Conference (MILCOM)*, Nov 2011, pp. 1720–1726.

[67] S. Vigneshwaran, I. Muthumani, and A. Raja, "Investigations on free space optics communication system," in *Information Communication and Embedded Systems (ICICES), 2013*, Feb 2013, pp. 819–824.

[68] H. Le-Minh and Z. Ghassemlooy and M. Ijaz and S. Rajbhandari and O. Adebanjo and S. Ansari and E. Leitgeb, "Experi-

mental study of bit error rate of free space optics communications in laboratory controlled turbulence," in *IEEE Globecom Workshops*, Dec 2010, pp. 1072–1076.

[69] F. Demers, H. Yanikomeroglu, and M. St-Hilaire, "A survey of opportunities for free space optics in next generation cellular networks," in *Communication Networks and Services Research Conference (CNSR), 2011 Ninth Annual*, May 2011, pp. 210–216.

[70] M. Feng, J. Wang, M. Sheng, L. Cao, X. Xie and M. Chen, "Outage performance for parallel relay-assisted free-space optical communications in strong turbulence with pointing errors," in *International Conference on Wireless Communications and Signal Processing (WCSP)*, Nanjing, Nov 2011, pp. 1–5.

[71] Z. Guo and Y. Yang, "On nonblocking multicast fat-tree data center networks with server redundancy," *IEEE Transactions on Computers*, vol. 64, no. 4, pp. 1058–1073, April 2015.

[72] Z. Guo and Y. Yang, "Exploring server redundancy in nonblocking multicast data center networks," *IEEE Transactions on Computers*, vol. 64, no. 7, pp. 1912–1926, July 2015.

[73] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, "A link failure recovery algorithm for virtual network function chaining," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, May 2017, pp. 213–221.

[74] Y. Xu and V. P. Kafle, "Reliable service function chain provisioning in software-defined networking," in *2017 13th International Conference on Network and Service Management (CNSM)*, Nov 2017, pp. 1–4.

[75] S. Herker, X. An, W. Kiess, S. Beker, and A. Kirstaedter, "Data-center architecture impacts on virtualized network functions service chain embedding with high availability requirements," in *2015 IEEE Globecom Workshops (GC Wkshps)*, Dec 2015, pp. 1–7.

[76] L. Qu, C. Assi, K. Shaban, and M. Khabbaz, "Reliability-aware service provisioning in nfv-enabled enterprise datacenter networks," in *CNSM*, Oct 2016, pp. 153–159.

[77] W. Ding, H. Yu and S. Luo, "Enhancing the reliability of services in nfv with the cost-efficient redundancy scheme," in *IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.

[78] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, "Virtual network function placement for resilient service chain provisioning," in *RNDM*, Sept 2016, pp. 245–252.

[79] H. Ye, S. Ma, Z. Yu, Y. Liu, and Z. Chen, "Simultaneous all-optical or and xor logic gates based on the bimodal photonic cavity containing a quantum dot," *IEEE Photonics Journal*, vol. 8, no. 6, pp. 1–10, Dec 2016.

[80] Y.-S. Dai, Y. Pan, and X. Zou, "A hierarchical modeling and analysis for grid service reliability," *IEEE Trans. Comput.*, vol. 56, no. 5, pp. 681–691, May 2007. [Online]. Available: https://doi.org/10.1109/TC.2007.1034

[81] Y.-S. Dai, B. Yang, J. Dongarra, and G. Zhang, "Cloud service reliability: Modeling and analysis," *Semantic Scholar*, 2010.

[82] T. A. Nguyen, D. Min, E. Choi, and T. D. Tran, "Reliability and availability evaluation for cloud data center networks using hierarchical models," *IEEE Access*, vol. 7, pp. 9273–9313, 2019.

[83] "ETSI GS NFV 002 V1.1.1: Network Functions Visualisation (NFV); Architectural Framework."

[84] L. Peterson and S. Das, "Trellis: Cord network infrastructure," 2017. [Online]. Available: https://wiki.opencord.org/pages/viewpage.action?pageId=2557405

[85] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, A. Kanagala, J. Provost, J. Simmons, E. Tanda, J. Wanderer, U. Hölzle, S. Stuart, and A. Vahdat, "Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network," in *Sigcomm '15*, 2015.

[86] J. M. Halpern and C. Pignataro, "Service Function Chaining (SFC) Architecture," RFC 7665, Oct. 2015. [Online]. Available: https://rfc-editor.org/rfc/rfc7665.txt

[87] P. Quinn, U. Elzur, and C. Pignataro, "Network Service Header (NSH)," RFC 8300, Jan. 2018. [Online]. Available: https://rfc-editor.org/rfc/rfc8300.txt

[88] G. Singh et al., "ETSI GS NFV-REL 004 V1.1.1: Network Functions Visualisation (NFV); Assurance; Report on Active Monitoring and Failure Detection."

[89] M. Furdek, N. Skorin-Kapov, S. Zsigmond, and L. Wosinska, "Vulnerabilities and security issues in optical networks," in *Transparent Optical Networks (ICTON),*, July 2014, pp. 1–4.

[90] S. Yuan and D. Stewart, "Protection of optical networks against interchannel eavesdropping and jamming attacks," in *Computational Science and Computational Intelligence (CSCI),*, vol. 1, March 2014, pp. 34–38.

[91] M. Fok, Z. Wang, Y. Deng, and P. Prucnal, "Optical layer security in fiber-optic networks," *Information Forensics and Security, IEEE Transactions on*, vol. 6, no. 3, pp. 725–736, Sept 2011.

[92] S. Pfennig and E. Franz, "Secure network coding: Dependency of efficiency on network topology," in *Communications (ICC), 2013 IEEE International Conference on*, June 2013, pp. 2100–2105.

[93] J. Wang, J. Wang, K. Lu, B. Xiao, and N. Gu, "Modeling and optimal design of linear network coding for secure unicast with multiple streams," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 10, pp. 2025–2035, Oct 2013.

[94] J. Daemen and V. Rijmen, "Aes proposal: Rijndael," 1999.

[95] Computer Security Resource Center, "Announcing the ADVANCED ENCRYPTION STANDARD (AES)," NIST, 2001.

[96] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, Nov 1976.

[97] "ITU-T Recommendation G.709: Interfaces for the Optical Transport Network (OTN) ," ITU-T Recommendations.

[98] "ITU-T Recommendation G.7715/Y.1706 (2002), Architecture and Requirements for Routing in the Automatically Switched Optical Network," ITU-T Recommendations.

[99] A. Soltani and S. Sharifian, "An ultra-high throughput and fully pipelined implementation of aes algorithm on fpga," *Microprocess. Microsyst.*, vol. 39, no. 7, pp. 480–493, Oct. 2015.

[100] G. Cincotti, N. Wada, and K. i. Kitayama, "Secure optical bit- and block-cipher transmission using a single multiport encoder/decoder," in *OFC/NFOEC*, Feb 2008, pp. 1–3.

[101] Z. Pan and L. Zhang, "Optical cryptography-based temporal ghost imaging with chaotic laser," *IEEE Photonics Technology Letters*, vol. 29, no. 16, pp. 1289–1292, Aug 2017.

[102] V. W. S. Chan, "Classical optical cryptography," in *17th International Conference on Transparent Optical Networks (ICTON)*, Budapest 2015, pp. 1–4.

[103] S. Donati and V. Annovazzi-Lodi, "From order to chaos and back: Recent advances in optical cryptography of transmitted data," in *CAOL*, Sept 2013, pp. 1–6.

[104] M. L. F. Abbade, M. Cvijetic, C. A. Messani, C. J. Alves and S. Tenenbaum, "Double all-optical encryption of m-qam signals based on spectrally sliced encoding keys," in *ICTON*, 2015, pp. 1–4.

[105] K. Mpalane, N. Gasela, B. M. Esiefarienrhe, and H. D. Tsague, "Vulnerability of advanced encryption standard algorithm to differential power analysis attacks implemented on atmega-128 microcontroller," in *AIPR*, Sept 2016, pp. 1–5.

[106] N. Smart, *Algorithms, Key Size and Parameters: Report - 2014*, European Network and Information Security Agency.

[107] E. B. Barker and J. M. Kelsey, "Sp 800-90a. recommendation for random number generation using deterministic random bit generators," Tech. Rep., 2015.

[108] ISO/IEC 18031:2011, "Information technology - Security techniques - Random bit generation," 2011.

[109] K. Kitayama and M. Sasaki and S. Araki and M. Tsubokawa and A. Tomita and K. Inoue and K. Harasawa and Y. Nagasako and A. Takada, "Security in photonic networks: Threats and security enhancement," *J. Light. Technol.*, vol. 29, no. 21, pp. 3210–3222, Nov 2011.

[110] A. Aguado and V. López and J. Marhuenda and O. González de Dios and J. P. Fernández-Palacios, "ABNO: a feasible SDN approach for multivendor IP and optical networks [invited]," *IEEE J. Opt. Commun. Netw.*, vol. 7, no. 2, pp. A356–A362, February 2015.

[111] M. Kodialam, T. V. Lakshman, J. B. Orlin, and S. Sengupta, "Oblivious routing of highly variable traffic in service overlays and ip backbones," *IEEE/ACM Trans. Netw.*, vol. 17, no. 2, pp. 459–472, Apr. 2009.

[112] A. M. Gergely and B. Crainicu, "A succinct survey on (pseudo)-random number generators from a cryptographic perspective," in *ISDFS*, April 2017, pp. 1–6.

[113] C. E. Shannon, "Communication theory of secrecy systems," *The Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[114] G. Vernam, "Secret signaling system," US Patent 1,310,719, Google Patents, 1919.

[115] S. Goldwasser and S. Micali, "Probabilistic encryption." *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[116] A. Russell and H. Wang, *How to fool an unbounded adversary with a short key.* Springer Berlin Heidelberg.

[117] Y. Dodis and A. Smith, "Entropic security and the encryption of high entropy messages," in *Proceedings of the Second International Conference on Theory of Cryptography*, ser. TCC'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 556–577.

[118] "eSTREAM: the ECRYPT Stream Cipher Project," 2012. [Online]. Available: http://www.ecrypt.eu.org/stream/

[119] C. Cid, M. Robshaw, S. Babbage, J. Borghoff, and V. Velichkov, "The estream portfolio in 2012." ECRYPT II, 2012.

[120] H. Wu, *The Stream Cipher HC-128*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 39–47.

[121] C. Berbain, O. Billet, A. Canteaut, N. Courtois, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin and H. Sibert, *Sosemanuk, a Fast Software-Oriented Stream Cipher*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 98–118.

[122] M. Boesgaard, M. Vesterager, and E. Zenner, *The Rabbit Stream Cipher*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 69–83.

[123] S. Kaur, R.-S. Kaler, and T.-S. Kamal, "All-optical binary full adder using logic operations based on the nonlinear properties of a semiconductor optical amplifier," *J. Opt. Soc. Korea*, vol. 19, no. 3, pp. 222–227.

[124] D. J. Bernstein, *The Salsa20 Family of Stream Ciphers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 84–97.

[125] M. Agren, M. Hell, T. Johansson, and W. Meier, "Grain-128a: A new version of grain-128 with optional authentication," *Int. J. Wire. Mob. Comput.*, vol. 5, no. 1, pp. 48–59, Dec. 2011.

[126] S. Babbage and M. Dodd, *The MICKEY Stream Ciphers*. Springer Berlin Heidelberg, 2008, pp. 191–209.

[127] C. De Cannière, "Trivium: A stream cipher construction inspired by block cipher design principles," in *Information Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 171–186.

[128] J. M. Marmolejo-Tejada, V. Trujillo-Olaya, and J. Velasco-Medina, "Hardware implementation of grain-128, mickey-128, decim-128 and trivium," in *2010 IEEE ANDESCON*, Sept 2010, pp. 1–6.

[129] X. Zhang et. al, "High-speed all-optical encryption and decryption based on two-photon absorption in semiconductor optical amplifiers," *IEEE/OSA JOCN*, vol. 7, no. 4, pp. 276–285, 2015.

[130] A. Bhardwaj and J. Jaques, *All-optical linear feedback shift register*. US Patent App. 10/804,321, Google Patents, 2005.

[131] M. Scaffardi, G. Berrettini, A. T. Nguyen, F. Bontempi, and A. Bogoni, "Optical linear feedback shift register," in *2011 Conference on Lasers and Electro-Optics Europe and 12th European Quantum Electronics Conference (CLEO EUROPE/EQEC)*, 2011, pp. 1–1.

[132] S. Even and O. Goldreich, "On the power of cascade ciphers," *ACM Trans. Comput. Syst.*, vol. 3, no. 2, p. 108–116, May 1985. [Online]. Available: https://doi.org/10.1145/214438.214442

[133] S. Chow, P. Eisen, H. Johnson, and P. C. van Oorschot, "A white-box DES implementation for DRM applications," in *Digital Rights Management*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 1–15.

[134] S. Chow, P. Eisen, H. Johnson, and P. C. Van Oorschot, "White-box cryptography and an AES implementation," in *Selected Areas in Cryptography*, K. Nyberg and H. Heys, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 250–270.

[135] J. Daemen and V. Rijmen, *The Design of Rijndael*. Berlin, Heidelberg: Springer-Verlag, 2002.

[136] A. Marghescu, P. Svasta, and E. Simion, "High speed and secure variable probability pseudo/true random number generator using fpga," in *2015 IEEE 21st International Symposium for Design and Technology in Electronic Packaging (SIITME)*, 2015, pp. 323–328.

Acronyms

ACF	Acceptable Component Failure
AES	Advanced Encryption Standard
APL	Acceptable Packet Loss
BER	Bit Error Rate
CR	Core Router
cTor	mTor with Coding
DC	Data Center
DCN	Data Center Network
ECMP	Equal-Cost Multi-Path
FCSR	Feedback with Carry Shift Register
FCT	Flow Completion Time
FEC	Forward Error Correction
FSO	Free Space Optics
FW	Firewall
GCH	Generalized Coding Header
GFP	Generic Framing Procedure
IDS	Intrusion Detection System
LB	Load Balancer
LFSR	Linear Feedback Shift Register
LNC	Linear Network Coding
MDS	Maximum Distance Separable
MLD	Multi-Lane Distribution
mTor	Multiple-Circuit Tor
NAT	Network Address Translation

NFV	Network Function Virtualization
NLFSR	Non-Linear Feedback Shift Registers
ODU	Optical Data Unit
oKG	optical Key Generator
OP	Onion Proxy
OPS	Optical Packet Switching
OR	Onion Router
oSC	optical Stream Cipher
OSNR	Optical Signal to Noise Ratio
OTN	Optical Transport Networks
PCS	Physical Coding Sublayer
PER	Packet Error Rate
pRNG	pseudo-Random Number Generator
QoS	Quality of Service
RF	Radio-Frequency
RLNC	Random Linear Network Coding
SDN	Software Defined Network
SER	Symbol Error Rate
SERG	Symbol Error Rate Guard
SFC	Service Function Chain
SFP	Service Function Path
SR	Segment Routing
ToR	Top-of-Rack
Tor	The Onion Routing
TSp	Traffic Shaper
VM	Virtual Machine
VNF	Virtual Network Functions
VOQ	Virtual Output Queues
vS	Virtual Switch
WDM	Wavelength Division Multiplexing